

The l3backend-testphase package

Additional backend PDF features

L^AT_EX PDF management testphase bundle

The L^AT_EX Project*

Version 0.95g, released 2021-07-21

1 l3backend-testphase Implementation

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

1.1 Crossreferences

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

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

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

```

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

```

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

```

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

```

```

\g__pdf_tmpa_prop
\l__pdf_tmpa_tl
\l__pdf_backend_tmpa_box

```

Some scratch variables

```

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

```

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

```

\g__pdf_backend_resourceid_int
\g__pdf_backend_name_int
\g__pdf_backend_page_int

```

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

```

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

```

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

1.2 luacode

Load the lua code.

```

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

```

1.3 Converting unicode strings to a pdfname

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

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

1.4 Hooks

1.4.1 Add the “end run” hooks

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

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

1.4.2 Add the “shipout” hooks

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

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

```

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

```

1.5 The /Pages dictionary (pdfpagesattr)

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

```

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

```

145 `</dvisvgm>`

(End definition for `_pdf_backend_Pages_primitive:n`.)

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

```

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

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

```

```

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

```

```

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

```

```

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

```



```

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

```

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

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

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

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

```

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

```

```

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

```

(End definition for \c__pdf_backend_PageResources_clist.)

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

__pdf_backend_PageResources_gput:nnn stores values for the page resources.

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

#2 : a pdf name without slash

#3 : value

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

__pdf_backend_PageResources_obj_gpush:

```

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

```

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

```

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

```

code for end of document code

```

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

```

```

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

```

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

```

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

```

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

```

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

```

```

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

```

dvipsvgm unneeded, or no-op

```

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

```

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

1.7.1 Page resources /Properties + BDC operators

```

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

```

```

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

```

```

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

```

```

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

```

```

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

```



```

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

```

```

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

```

842 % pdflatex
843 <*pdftex>
844 \cs_new_protected:Npn \_pdf_backend_NamesEmbeddedFiles_gpush:n #1 %array content
845 {
846     \pdf_object_unnamed_write:nn {dict} {/Names [#1] }
847     \tex_pdfnames:D {/EmbeddedFiles~\pdf_object_ref_last:}
848 }
849 </pdftex>
850 <*luatex>
851 \cs_new_protected:Npn \_pdf_backend_NamesEmbeddedFiles_gpush:n #1 %array content
852 {
853     \pdf_object_unnamed_write:nn {dict} {/Names [#1] }
854     \tex_pdfextension:D-names~{/EmbeddedFiles~\pdf_object_ref_last: }
855 }
856 </luatex>
857 <*dviPDFmx | xdvipdfmx>
858 \cs_new_protected:Npn \_pdf_backend_NamesEmbeddedFiles_gpush:n #1 %array content
859 {
860     \pdf_object_unnamed_write:nn {dict} { /Names [#1] }
861     %n or x?
862     \_pdf_backend:x {put~@names~<</EmbeddedFiles~\pdf_object_ref_last: >>}
863 }
864 </dviPDFmx | xdvipdfmx>
865
866 %dvips: noop
867 <*dvips>
868 \cs_new_protected:Npn \_pdf_backend_NamesEmbeddedFiles_gpush:n #1 {}
869 </dvips>
870 %dvisvgm: noop
871 <*dvisvgm>
872 \cs_new_protected:Npn \_pdf_backend_NamesEmbeddedFiles_gpush:n #1 {}
873 </dvisvgm>
874

```

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:`

```

875 <*drivers>
876 \int_new:N \g__pdf_backend_EmbeddedFiles_int
877 \cs_new:Npn \_pdf_backend_EmbeddedFiles_name:
878 {
879   (
880     l3ef
881     \int_compare:nNtT {\g__pdf_backend_EmbeddedFiles_int} < {10}
882     {0}
883     \int_compare:nNtT {\g__pdf_backend_EmbeddedFiles_int} < {100}
884     {0}
885     \int_compare:nNtT {\g__pdf_backend_EmbeddedFiles_int} < {1000}
886     {0}
887     \int_use:N \g__pdf_backend_EmbeddedFiles_int
888   )
889 }
890 </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`.)

```

891 <*drivers>
892 \seq_new:N \g__pdf_backend_EmbeddedFiles_seq
893 \prop_new:N \g__pdf_backend_EmbeddedFiles_named_prop
894 </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.

```

895 <*pdfTeX | luatex | dvipdfmx | xdvipdfmx>
896 \cs_new_protected:Npn \_pdf_backend_NamesEmbeddedFiles_add:n #1
897   %#1 object ref
898   {
899     \int_gincr:N \g__pdf_backend_EmbeddedFiles_int
900     \prop_gput:Nnx \g__pdf_backend_EmbeddedFiles_named_prop
901       { #1 }
902     { \_pdf_backend_EmbeddedFiles_name: }
903     \seq_gput_right:Nx \g__pdf_backend_EmbeddedFiles_seq
904       { \_pdf_backend_EmbeddedFiles_name: \c_space_tl #1 }
905   }
906
907 </pdfTeX | luatex | dvipdfmx | xdvipdfmx>
908 <*dvips>
909 \cs_new_protected:Npn \_pdf_backend_NamesEmbeddedFiles_add:n #1
910   {
911     \int_gincr:N \g__pdf_backend_EmbeddedFiles_int
912     \prop_gput:Nnx \g__pdf_backend_EmbeddedFiles_named_prop
913       { #1 }

```

```

914         { \_pdf_backend_EmbeddedFiles_name: }
915     \_pdf_backend_pdfmark:x
916     {
917         /Name~\_pdf_backend_EmbeddedFiles_name:~
918         /FS~#1~
919         /EMBED
920     }
921 }
922 </dvips>
923 <*dvisvgm>
924 %no op. Or is there any sensible use for it?
925 \cs_new_protected:Npn \_pdf_backend_NamesEmbeddedFiles_add:n #1
926 {}
927 </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 928 <*pdftex>
\_pdf_backend_xform_ref:n 929 \cs_new_protected:Npn \_pdf_backend_xform_new:nnnn #1 #2 #3 #4
                             930 % #1 name
                             931 % #2 attributes
                             932 % #3 resources
                             933 % #4 content, not necessarily a box!
                             934 {
                             935     \hbox_set:Nn \l__pdf_backend_tmpa_box
                             936     {
                             937         \bool_set_true:N \l__pdf_backend_xform_bool
                             938         \prop_gc_clear:c {\_kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties }}
                             939         #4
                             940     }
                             941     %store the dimensions
                             942     \tl_const:cx
                             943     { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
                             944     { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
                             945     \tl_const:cx
                             946     { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
                             947     { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
                             948     \tl_const:cx
                             949     { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
                             950     { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
                             951     %% do we need to test if #2 and #3 are empty??
                             952     \tex_immediate:D \tex_pdfxform:D
                             953     ~ attr ~ { #2 }
                             954     %% which other resources should be default? Is an argument actually needed?
                             955     ~ resources ~
                             956     {

```

```

957     #3
958     \int_compare:nNnT
959       { \prop_count:c { \__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties
960         >
961         { 0 }
962         {
963           /Properties~
964           <<
965           \pdffdict_use:n { g__pdf_Core/Xform/Resources/Properties }
966           >>
967         }
968
969     \prop_if_empty:cF
970     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
971     {
972       /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
973     }
974     \prop_if_empty:cF
975     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
976     {
977       /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
978     }
979     \prop_if_empty:cF
980     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Shading } }
981     {
982       /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
983     }
984     \prop_if_empty:cF
985     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
986     {
987       /ColorSpace~ \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
988     }
989   }
990   \l__pdf_backend_tmpa_box
991   \int_const:cn
992   { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
993   { \tex_pdflastxform:D }
994 }
995
996 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
997 {
998   \tex_pdfrefxform:D
999   \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1000   \scan_stop:
1001 }
1002
1003 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1004 {
1005   \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R
1006 }
1007 </pdfTeX>
1008 <*luatex>
1009 %luatex
1010 %nearly identical but not completely ...

```

```

1011 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
1012 % #1 name
1013 % #2 attributes
1014 % #3 resources
1015 % #4 content, not necessarily a box!
1016 {
1017   \hbox_set:Nn \l__pdf_backend_tmpa_box
1018   {
1019     \bool_set_true:N \l__pdf_backend_xform_bool
1020     \prop_gclear:c { \__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties }
1021     #4
1022   }
1023   \tl_const:cx
1024   { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1025   { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1026   \tl_const:cx
1027   { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1028   { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1029   \tl_const:cx
1030   { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1031   { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1032   %% do we need to test if #2 and #3 are empty??
1033   \tex_immediate:D \tex_pdfxform:D
1034   ~ attr ~ { #2 }
1035   %% which resources should be default? Is an argument actually needed?
1036   ~ resources ~
1037   {
1038     #3
1039     \int_compare:nNnT
1040     { \prop_count:c { \__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties
1041     >
1042     { 0 }
1043     {
1044       /Properties~
1045       <<
1046       \pdffdict_use:n { g__pdf_Core/Xform/Resources/Properties }
1047       >>
1048     }
1049     \prop_if_empty:cF
1050     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
1051     {
1052       /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1053     }
1054     \prop_if_empty:cF
1055     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
1056     {
1057       /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1058     }
1059     \prop_if_empty:cF
1060     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Shading } }
1061     {
1062       /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1063     }
1064     \prop_if_empty:cF

```

```

1065         { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
1066         {
1067             /ColorSpace~ \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1068         }
1069     }
1070     \l__pdf_backend_tmpa_box
1071     \int_const:cn
1072     { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1073     { \tex_pdflastxform:D }
1074 }
1075
1076 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 %protected as with xelatex
1077 {
1078     \tex_pdfrefxform:D \int_use:c
1079     {
1080         c__pdf_backend_xform_ \tl_to_str:n {#1} _int
1081     }
1082     \scan_stop:
1083 }
1084
1085 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1086 { \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R }
1087
1088 </luatex>
1089 <*dvipdfmx | xdvipdfmx>
1090 % xetex
1091 % it needs a bit testing if it really works to set the box to 0 before the special ...
1092 % does it disturb viewing the xobject?
1093 % what happens with the resources (bdc)? (should work as they are specials too)
1094 % xetex requires that the special is in horizontal mode. This means it affects
1095 % typesetting. But we can no delay the whole form code to shipout
1096 % as the object reference and the size is often wanted on the current page.
1097 % so we need to allocate a box - but probably they won't be thousands xform
1098 % in a document so it shouldn't matter.
1099 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
1100 % #1 name
1101 % #2 attributes
1102 % #3 resources
1103 % #4 content, not necessarily a box!
1104 {
1105     \int_gincr:N \g__pdf_backend_object_int
1106     \int_const:cn
1107     { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1108     { \g__pdf_backend_object_int }
1109     \box_new:c { g__pdf_backend_xform_#1_box }
1110     \hbox_gset:cn { g__pdf_backend_xform_#1_box }
1111     {
1112         \bool_set_true:N \l__pdf_backend_xform_bool
1113         #4
1114     }
1115     \tl_const:cx
1116     { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1117     { \tex_the:D \box_wd:c { g__pdf_backend_xform_#1_box } }
1118     \tl_const:cx

```



```

1119 { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1120 { \tex_the:D \box_ht:c { g__pdf_backend_xform_#1_box } }
1121 \tl_const:cx
1122 { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1123 { \tex_the:D \box_dp:c { g__pdf_backend_xform_#1_box } }
1124 \box_set_dp:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1125 \box_set_ht:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1126 \box_set_wd:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1127 \hook_gput_next_code:nn {shipout/background}
1128 {
1129   \mode_leave_vertical: %needed, the xform disappears without it.
1130   \__pdf_backend:x
1131   {
1132     bxobj ~ \__pdf_backend_xform_ref:n { #1 }
1133     \c_space_tl width ~ \pdfxform_wd:n { #1 }
1134     \c_space_tl height ~ \pdfxform_ht:n { #1 }
1135     \c_space_tl depth ~ \pdfxform_dp:n { #1 }
1136   }
1137   \box_use_drop:c { g__pdf_backend_xform_#1_box }
1138   \__pdf_backend:x {put ~ @resources ~<<#3>> }
1139   \__pdf_backend:x
1140   {
1141     put~ @resources ~
1142     <<
1143     /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1144     >>
1145   }
1146   \__pdf_backend:x
1147   {
1148     put~ @resources ~
1149     <<
1150     /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1151     >>
1152   }
1153   \__pdf_backend:x
1154   {
1155     put~ @resources ~
1156     <<
1157     /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1158     >>
1159   }
1160   \__pdf_backend:x
1161   {
1162     put~ @resources ~
1163     <<
1164     /ColorSpace~
1165     \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1166     >>
1167   }
1168   \exp_args:Nx
1169   \__pdf_backend:x {exobj ~<<#2>>}
1170 }
1171 }
1172

```

```

1173
1174
1175 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1176 {
1177   @pdf.xform \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1178 }
1179
1180 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1181 {
1182   \hbox_set:Nn \l__pdf_backend_tmpa_box
1183   {
1184     \__pdf_backend:x
1185     {
1186       uxobj~ \__pdf_backend_xform_ref:n { #1 }
1187     }
1188   }
1189   \box_set_wd:Nn \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1190   \box_set_ht:Nn \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1191   \box_set_dp:Nn \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1192   \box_use_drop:N \l__pdf_backend_tmpa_box
1193 }
1194 </dviptfm> | xdvipdfmx>
1195 <*dvism>
1196 % unclear what it should do!!
1197 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4 {}
1198 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 {}
1199 \cs_new:Npn \__pdf_backend_xform_ref:n {}
1200 </dvism>

```

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.

```

1201 <*dvips>
1202 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4 % #1 name, #2 attribute, #4
1203 {
1204   \int_gincr:N \g__pdf_backend_object_int
1205   \int_const:cn
1206   { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1207   { \g__pdf_backend_object_int }
1208
1209   \hbox_set:Nn \l__pdf_backend_tmpa_box
1210   {
1211     \bool_set_true:N \l__pdf_backend_xform_bool
1212     \prop_gclear:c {\__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties }}
1213     #4
1214   }
1215   %store the dimensions
1216   \tl_const:cx
1217   { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1218   { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1219   \tl_const:cx
1220   { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1221   { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1222   \tl_const:cx

```

```

1223     { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1224     { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1225 %mirror the box
1226 \box_scale:Nnn \l__pdf_backend_tmpa_box {1} {-1}
1227 \hbox_set:Nn\l__pdf_backend_tmpb_box
1228 {
1229     \__kernel_backend_postscript:x
1230     {
1231         gsave~currentpoint~translate~
1232         mark~
1233         /_objdef~{ pdf.obj \int_use:N\g__pdf_backend_object_int }\c_space_tl~
1234         /BBox[
1235         currentpoint~\dim_to_decimal:n{\box_dp:N \l__pdf_backend_tmpa_box}~72.27~div~VRe
1236         currentpoint~
1237         exch~\dim_to_decimal:n{\box_wd:N \l__pdf_backend_tmpa_box}~72.27~div~Resolutio
1238         exch~\dim_to_decimal:n{\box_ht:N \l__pdf_backend_tmpa_box}~72.27~div~VResolutio
1239         ]
1240         \str_if_eq:eeF{#1}{}
1241         {
1242             product~(Distiller)~search~{pop~pop~pop~#2}{pop}ifelse~
1243             }
1244         /BP~pdfmark~grestore~
1245         }
1246     \box_use_drop:N\l__pdf_backend_tmpa_box
1247     \__kernel_backend_postscript:n
1248     {
1249         mark ~ /EP~pdfmark
1250     }
1251     \str_if_eq:eeF{#1}{}
1252     {
1253         \__kernel_backend_postscript:x
1254         {
1255             product~(Ghostscript)~search~
1256             {
1257                 pop~pop~pop~
1258                 mark~
1259                 { pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }
1260                 ~<<#2>>~/PUT~pdfmark
1261                 }{pop}ifelse
1262             }
1263         }
1264     }
1265     \box_set_dp:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1266     \box_set_ht:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1267     \box_set_wd:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1268     \hook_gput_code:nnn {begindocument/end}{pdfxform}
1269     {
1270         \mode_leave_vertical:
1271         \box_use:N\l__pdf_backend_tmpb_box
1272     }
1273 }
1274
1275
1276 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1

```

```

1277 {
1278   \hbox_set:Nn \l__pdf_backend_tmpa_box
1279   {
1280     \__kernel_backend_postscript:x
1281     {
1282       gsave~currentpoint~translate~1~-1~scale~
1283       mark~{ pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int }}~
1284       /SP~pdfmark ~ grestore
1285     }
1286   }
1287   \box_set_wd:Nn \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1288   \box_set_ht:Nn \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1289   \box_set_dp:Nn \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1290   \box_use_drop:N \l__pdf_backend_tmpa_box
1291 }
1292 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1293 {
1294   { pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }
1295 }
1296
1297 </dvips>
1298 <*drivers>
1299 %% all
1300 \prg_new_conditional:Npnn \__pdf_backend_xform_if_exist:n #1 { p , T , F , TF }
1301 {
1302   \int_if_exist:cTF { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1303   { \prg_return_true: }
1304   { \prg_return_false: }
1305 }
1306 \prg_new_eq_conditional:NNn \pdfxform_if_exist:n \__pdf_backend_xform_if_exist:n
1307 { TF , T , F , p }
1308 </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

```

1309 <*lua>
1310 ltx= ltx or {}
1311 ltx.__pdf = ltx.__pdf or {}
1312 ltx.__pdf.Page = ltx.__pdf.Page or {}
1313 ltx.__pdf.Page.dflt = ltx.__pdf.Page.dflt or {}
1314 ltx.__pdf.Page.Resources = ltx.__pdf.Page.Resources or {}
1315 ltx.__pdf.Page.Resources.Properties = ltx.__pdf.Page.Resources.Properties or {}
1316 ltx.__pdf.Page.Resources.List={"ExtGState","ColorSpace","Pattern","Shading"}
1317 ltx.__pdf.object = ltx.__pdf.object or {}
1318
1319 ltx.pdf= ltx.pdf or {} -- for "public" functions
1320
1321 local __pdf = ltx.__pdf
1322 local pdf = pdf
1323
1324 local function __pdf_backend_Page_gput (name,value)

```

```

1325 __pdf.Page.dflt[name]=value
1326 end
1327
1328 local function __pdf_backend_Page_gremove (name)
1329 __pdf.Page.dflt[name]=nil
1330 end
1331
1332 local function __pdf_backend_Page_gclear ()
1333 __pdf.Page.dflt={}
1334 end
1335
1336 local function __pdf_backend_ThisPage_gput (page,name,value)
1337 __pdf.Page[page] = __pdf.Page[page] or {}
1338 __pdf.Page[page][name]=value
1339 end
1340
1341 local function __pdf_backend_ThisPage_gpush (page)
1342 local token=""
1343 local t = {}
1344 local tkeys= {}
1345 for name,value in pairs(__pdf.Page.dflt) do
1346 t[name]=value
1347 end
1348 if __pdf.Page[page] then
1349 for name,value in pairs(__pdf.Page[page]) do
1350 t[name] = value
1351 end
1352 end
1353 -- sort the table to get reliable test files.
1354 for name,value in pairs(t) do
1355 table.insert(tkeys,name)
1356 end
1357 table.sort(tkeys)
1358 for _,name in ipairs(tkeys) do
1359 token = token .. "/"..name.." " ..t[name]
1360 end
1361 return token
1362 end
1363
1364 function ltx.__pdf.backend_ThisPage_gput (page,name,value) -- tex.count["g_shipout_readonly_
1365 __pdf_backend_ThisPage_gput (page,name,value)
1366 end
1367
1368 function ltx.__pdf.backend_ThisPage_gpush (page)
1369 pdf.setpageattributes(__pdf_backend_ThisPage_gpush (page))
1370 end
1371
1372 function ltx.__pdf.backend_Page_gput (name,value)
1373 __pdf_backend_Page_gput (name,value)
1374 end
1375
1376 function ltx.__pdf.backend_Page_gremove (name)
1377 __pdf_backend_Page_gremove (name)
1378 end

```

```

1379
1380 function ltx.__pdf.backend_Page_gclear ()
1381   __pdf_backend_Page_gclear ()
1382 end
1383
1384
1385 local Properties = ltx.__pdf.Page.Resources.Properties
1386 local ResourceList= ltx.__pdf.Page.Resources.List
1387 local function __pdf_backend_PageResources_gpush (page)
1388   local token=""
1389   if Properties[page] then
1390     -- we sort the table, so that the pdf test works
1391     local t = {}
1392     for name,value in pairs (Properties[page]) do
1393       table.insert (t,name)
1394     end
1395     table.sort (t)
1396     for _,name in ipairs(t) do
1397       token = token .. "/"..name.." " .. Properties[page][name]
1398     end
1399     token = "/Properties <<"..token..">>"
1400   end
1401   for i,name in ipairs(ResourceList) do
1402     if ltx.__pdf.Page.Resources[name] then
1403       token = token .. "/"..name.." " ..ltx.pdf.object_ref("__pdf/Page/Resources/"..name)
1404     end
1405   end
1406   return token
1407 end
1408
1409 -- the function is public, as I probably need it in tagpdf too ...
1410 function ltx.pdf.Page_Resources_Properties_gput (page,name,value) -- tex.count["g_shipout_re
1411   Properties[page] = Properties[page] or {}
1412   Properties[page][name]=value
1413   pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
1414 end
1415
1416 function ltx.pdf.Page_Resources_gpush(page)
1417   pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
1418 end
1419
1420 function ltx.pdf.object_ref (objname)
1421   if ltx.__pdf.object[objname] then
1422     local ref= ltx.__pdf.object[objname]
1423     return ref
1424   else
1425     return "false"
1426   end
1427 end
1428 </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

bool commands:
`\bool_if:NTF`
 ... 601, 634, 658, 692, 714, 753, 801
`\bool_new:N` 488
`\bool_set_true:N` 937, 1019, 1112, 1211
box commands:
`\box_dp:N` .. 950, 1031, 1123, 1224, 1235
`\box_ht:N` .. 947, 1028, 1120, 1221, 1238
`\box_new:N` 50, 51, 1109
`\box_scale:Nnn` 1226
`\box_set_dp:Nn` 1124, 1191, 1265, 1289
`\box_set_ht:Nn` 1125, 1190, 1266, 1288
`\box_set_wd:Nn` 1126, 1189, 1267, 1287
`\box_use:N` 1271
`\box_use_drop:N` 1137, 1192, 1246, 1290
`\box_wd:N` .. 944, 1025, 1117, 1218, 1237

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, 877, 1003, 1085, 1175, 1199, 1292
`\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, 868, 872,
 896, 909, 925, 929, 996, 1011, 1076,
 1099, 1180, 1197, 1198, 1202, 1276
`\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` ... 1235, 1237, 1238
`\c_zero_dim`
 .. 1124, 1125, 1126, 1265, 1266, 1267
`\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, 1168

H

hbox commands:
`\hbox_gset:Nn` 1110
`\hbox_set:Nn`
 ... 935, 1017, 1182, 1209, 1227, 1278
hook commands:
`\hook_gput_code:nnn` .. 104, 448, 1268
`\hook_gput_next_code:nn` 1127
`\hook_gset_rule:nnnn` 442, 443

I

int commands:
`\int_compare:nNnTF`
 ... 881, 883, 885, 958, 1039
`\int_const:Nn` .. 991, 1071, 1106, 1205
`\int_gincr:N` 175, 546, 565, 631, 655,
 709, 713, 748, 752, 899, 911, 1105, 1204
`\int_if_exist:NTF` 1302
`\int_new:N` 54, 55, 56, 876
`\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, 887, 999, 1005, 1078,
 1086, 1177, 1233, 1259, 1283, 1294

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 ..
..... 1229, 1247, 1253, 1280
\__kernel_pdf_name_from_unicode_-
e:n ..... 62, 68
\__kernel_pdffdict_name:n 201, 202,
204, 432, 458, 616, 814, 825, 830,
938, 959, 970, 975, 980, 985, 1020,
1040, 1050, 1055, 1060, 1065, 1212
\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: ... 1129, 1270

P
pdf commands:
\pdf_object_ref:n .....
816, 972, 977, 982, 987, 1052, 1057,
1062, 1067, 1143, 1150, 1157, 1165
\pdf_object_ref_last: . 847, 854, 862
\pdf_object_unnamed_write:nn ...
..... 590, 683, 792, 846, 853, 860
\pdf_object_write ..... 460
pdf internal commands:
\__pdf_backend:n .....
.. 139, 452, 461, 862, 1130, 1138,
1139, 1146, 1153, 1160, 1169, 1184
\__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
\g__pdf_backend_EmbeddedFiles_-
int ..... 875,
876, 881, 883, 885, 887, 899, 911
\__pdf_backend_EmbeddedFiles_-
name: ..... 875, 902, 904, 914, 917
\g__pdf_backend_EmbeddedFiles_-
named_prop .... 891, 893, 900, 912
\g__pdf_backend_EmbeddedFiles_-
seq ..... 891, 892, 903
\__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_NamesEmbeddedFiles_-
add:n ..... 895
\__pdf_backend_NamesEmbeddedFiles_-
gpush:n .... 844, 851, 858, 868, 872
\g__pdf_backend_object_int .....
..... 1105, 1108, 1204, 1207, 1233
\__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, 915
\__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, 935, 944, 947, 950, 990,

```


1017, 1025, 1028, 1031, 1070, 1182, 1189, 1190, 1191, 1192, 1209, 1218, 1221, 1224, 1226, 1235, 1237, 1238, 1246, 1278, 1287, 1288, 1289, 1290	\prop_new:N 48, 893
\l__pdf_backend_tmpb_box	\ProvidesExplFile 1
. . . . 51, 1227, 1265, 1266, 1267, 1271	
\l__pdf_backend_xform_bool	R
. 488, 634,	ref commands:
658, 714, 753, 937, 1019, 1112, 1211	\ref_label:nn . . . 29, 34, 177, 725, 764
__pdf_backend_xform_if_exist:n .	\ref_value:nn . . . 30, 39, 185, 733, 772
. 1300, 1306	\relax 97
__pdf_backend_xform_new:nnnn . . 928	\RequirePackage 28
__pdf_backend_xform_ref:n 928	
__pdf_backend_xform_use:n 928	S
\g__pdf_tmpa_prop . . . 47, 201, 206, 211	scan commands:
\l__pdf_tmpa_tl	\scan_stop: 1000, 1082
. . . . 47, 179, 188, 190, 193, 727,	seq commands:
736, 738, 741, 766, 775, 777, 780, 783	\seq_gput_right:Nn 903
pdfdict commands:	\seq_new:N 892
\pdfdict_gput:nnn	str commands:
. 158, 193, 295, 334, 370, 412, 423,	\str_convert_pdfname:n 64, 459
473, 481, 636, 660, 716, 740, 755, 779	\str_if_eq:nnTF 1240, 1251
\pdfdict_gremove:nn 165, 302, 341, 376	sys commands:
\pdfdict_if_exist:nTF . 188, 736, 775	\sys_if_engine luatex:TF 119
\pdfdict_item:nn 211, 816, 831	
\pdfdict_new:n 190, 738, 777	T
\pdfdict_show:n 783	T _E X and L ^A T _E X 2 _ε commands:
\pdfdict_use:n 321, 360, 436, 965, 1046	\@bsphack 33
pdfmanagement internal commands:	\@esphack 35
\g__pdfmanagement_active_bool . . .	\@kernel@after@enddocument@afterlastpage
. 601, 692, 801 74, 75
\pdfnames 19	\@kernel@after@shipout@background
\pdfpageref 2 95, 98
pdfxform commands:	\@kernel@after@shipout@lastpage .
\pdfxform_dp:n 1135, 1191, 1289 81, 82, 88, 89
\pdfxform_ht:n 1134, 1190, 1288	\@kernel@before@shipout@background
\pdfxform_if_exist:n 1306 97
\pdfxform_wd:n 1133, 1189, 1287	\g@addto@macro 97, 98
prg commands:	\zref@extractdefault . . 181, 729, 768
\prg_new_conditional:Npnn 1300	\zref@labelbylist 176, 722, 761
\prg_new_eq_conditional:NNn . . 1306	tex commands:
\prg_return_false: 1304	\tex_directlua:D 123, 234, 248, 398, 400
\prg_return_true: 1303	\tex_global:D 114, 151, 822
prop commands:	\tex_immediate:D 952, 1033
\prop_count:N 959, 1040	\tex_latelua:D
\prop_gclear:N 938, 1020, 1212 224, 260, 277, 413, 414, 642, 666
\prop_gput:Nnn 206, 458, 900, 912	\tex_luaescapestring:D 219
\prop_gset_eq:NN 201	\tex_pdfextension:D 854
\prop_if_empty:NTF	\tex_pdflastxform:D 993, 1073
. 431, 616, 813, 969,	\tex_pdfnames:D 847
774, 979, 984, 1049, 1054, 1059, 1064	\tex_pdfpageattr:D 151
\prop_if_exist:NTF 202, 824	\tex_pdfpageresources:D 822
\prop_map_function:NN 211, 829	\tex_pdfpagesattr:D 114
\prop_map_inline:Nn 204	\tex_pdfrefxform:D 998, 1078
	\tex_pdfxform:D 952, 1033
	\tex_special:D 133, 288, 327

<code>\tex_the:D</code> 942, 945, 948, 1023, 1026, 1029, 1115, 1118, 1121, 1216, 1219, 1222
.. 944, 947, 950, 1025, 1028, 1031, 1117, 1120, 1123, 1218, 1221, 1224	
<code>\tex_unexpanded:D</code>	219
text commands:	
<code>\text_expand:n</code>	64, 70
tl commands:	
<code>\c_space_tl</code>	943, 946, 949, 992, 999, 1005, 1024, 1027, 1030, 1072, 1080, 1086, 1107, 1116, 1119, 1122, 1177, 1206, 1217, 1220, 1223, 1259, 1283, 1294, 1302
.... 549, 557, 568, 576, 633, 657, 711, 750, 904, 1133, 1134, 1135, 1233	
<code>\tl_const:Nn</code>	
	<code>\tl_gput_right:Nn</code>
	75, 82, 89
	<code>\tl_if_exist:NTF</code>
	95
	<code>\tl_new:N</code>
	49
	<code>\tl_set:Nn</code>
	179, 727, 766
	<code>\tl_to_str:n</code>