

# The l3backend-testphase package Additional backend PDF features L<sup>A</sup>T<sub>E</sub>X PDF management testphase bundle

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

Version 0.95i, released 2021-08-28

## 1 l3backend-testphase Implementation

```
1 <drivers>\ProvidesExplFile
2 <*dvipdfmx>
3   {l3backend-testphase-dvipdfmx.def}{2021-08-28}{}
4   {LaTeX-PDF-management-testphase-bundle-backend-support: dvipdfmx}
5 </dvipdfmx>
6 <*dvips>
7   {l3backend-testphase-dvips.def}{2021-08-28}{}
8   {LaTeX-PDF-management-testphase-bundle-backend-support: dvips}
9 </dvips>
10 <*dvisvgm>
11   {l3backend-testphase-dvisvgm.def}{2021-08-28}{}
12   {LaTeX-PDF-management-testphase-bundle-backend-support: dvisvgm}
13 </dvisvgm>
14 <*luatex>
15   {l3backend-testphase-luatex.def}{2021-08-28}{}
16   {LaTeX-PDF-management-testphase-bundle-backend-support: PDF output (LuaTeX)}
17 </luatex>
18 <*pdftex>
19   {l3backend-testphase-pdftex.def}{2021-08-28}{}
20   {LaTeX-PDF-management-testphase-bundle-backend-support: PDF output (pdfTeX)}
21 </pdftex>
22 <*xdvipdfmx>
23   {l3backend-testphase-xetex.def}{2021-08-28}{}
24   {LaTeX-PDF-management-testphase-bundle-backend-support: XeTeX}
25 </xdvipdfmx>
```

### 1.1 Crossreferences

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

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

---

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

```

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

```

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

```

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

```

```

\g__pdf_tmpa_prop   Some scratch variables
  \l__pdf_tmpa_tl
\l__pdf_backend_tmpa_box
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_tmpb_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  a counter to create labels for the resources, a counter to number properties in bdc marks,
\g__pdf_backend_name_int       a counter for the \pdfpageref implementation.
\g__pdf_backend_page_int
53 <*drivers>
54 \int_new:N \g__pdf_backend_resourceid_int
55 \int_new:N \g__pdf_backend_name_int
56 \int_new:N \g__pdf_backend_page_int
57 </drivers>

```

*(End definition for \g\_\_pdf\_backend\_resourceid\_int, \g\_\_pdf\_backend\_name\_int, and \g\_\_pdf\_backend\_page\_int.)*

## 1.2 luacode

Load the lua code.

```

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

```

### 1.3 Converting unicode strings to a pdfname

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

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

### 1.4 Hooks

#### 1.4.1 Add the “end run” hooks

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

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

#### 1.4.2 Add the “shipout” hooks

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

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

```

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

```

## 1.5 The /Pages dictionary (pdfpagesattr)

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

```

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

```

```
145 </dvisvgm>
```

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

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

```
\_pdf_backend_Page_primitive:n \_pdf_backend_Page_primitive:n is the primitive command to add something to the
\_pdf_backend_Page_gput:nn /Page dictionary. It works differently for the backends: pdftex and luatex overwrite
\_pdf_backend_Page_gremove:n existing content, dvips and dvipdfmx are additive. luatex sets it in lua. The higher
\_pdf_backend_ThisPage_gput:nn level code has to take this into account. \_pdf_backend_Page_gput:nn stores default
\_pdf_backend_ThisPage_gpush:n values. \_pdf_backend_Page_gremove:n allows to remove a value. \_pdf_backend_
ThisPage_gput:nn adds a value to the current page. \_pdf_backend_ThisPage_
gpush:n merges the default and the current page values and add them to the dictionary
of the current page in \g__pdf_backend_thispage_shipout_tl.

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 % \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:mn {__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_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/Propert
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 <*dviptfm | 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 </dviptfm | 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 | dviptfm | xdvipdfmx>
```

```

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

```

(End definition for `\__pdf_backend_NamesEmbeddedFiles_add:nn`.)

## 1.8.2 Additional annotation commands

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

```

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

```

```

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

```

### 1.8.3 FormXObject / backend

```

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

```

```

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

```

```

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

```

```

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

```

```

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

```



```

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

```

```

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

```

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

```

1206 <*dvips>
1207 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4 % #1 name, #2 attribute, #4
1208 {
1209     \int_gincr:N \g__pdf_backend_object_int
1210     \int_const:cn
1211     { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1212     { \g__pdf_backend_object_int }
1213
1214     \hbox_set:Nn \l__pdf_backend_tmpa_box
1215     {
1216         \bool_set_true:N \l__pdf_backend_xform_bool
1217         \prop_gc clear:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties } }
1218         #4
1219     }
1220 %store the dimensions
1221     \tl_const:cx
1222     { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1223     { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1224     \tl_const:cx
1225     { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1226     { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1227     \tl_const:cx
1228     { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1229     { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1230 %mirror the box

```

```

1231 \box_scale:Nnn \l__pdf_backend_tmpa_box {1} {-1}
1232 \hbox_set:Nn\l__pdf_backend_tmpb_box
1233 {
1234   \__kernel_backend_postscript:x
1235   {
1236     gsave~currentpoint~translate~
1237     mark~
1238     /_objdef~{ pdf.obj \int_use:N\g__pdf_backend_object_int }\c_space_tl~
1239     /BBox[
1240     currentpoint~\dim_to_decimal:n{\box_dp:N \l__pdf_backend_tmpa_box}~72.27~div~VR
1241     currentpoint~
1242     exch~\dim_to_decimal:n{\box_wd:N \l__pdf_backend_tmpa_box}~72.27~div~Resolutio
1243     exch~\dim_to_decimal:n{\box_ht:N \l__pdf_backend_tmpa_box}~72.27~div~VResolutio
1244     ]
1245     \str_if_eq:eeF{#1}{}
1246     {
1247       product~(Distiller)~search~{pop~pop~pop~#2}{pop}ifelse~
1248     }
1249     /BP~pdfmark~grestore~
1250   }
1251   \box_use_drop:N\l__pdf_backend_tmpa_box
1252   \__kernel_backend_postscript:n
1253   {
1254     mark ~ /EP~pdfmark
1255   }
1256   \str_if_eq:eeF{#1}{}
1257   {
1258     \__kernel_backend_postscript:x
1259     {
1260       product~(Ghostscript)~search~
1261       {
1262         pop~pop~pop~
1263         mark~
1264         { pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }
1265         ~<<#2>>~/PUT~pdfmark
1266       }{pop}ifelse
1267     }
1268   }
1269 }
1270 \box_set_dp:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1271 \box_set_ht:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1272 \box_set_wd:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1273 \hook_gput_code:nnn {begindocument/end}{pdfxform}
1274 {
1275   \mode_leave_vertical:
1276   \box_use:N\l__pdf_backend_tmpb_box
1277 }
1278 }
1279
1280
1281 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1282 {
1283   \hbox_set:Nn \l__pdf_backend_tmpa_box
1284   {

```

```

1285     \__kernel_backend_postscript:x
1286     {
1287         gsave~currentpoint~translate~1~-1~scale~
1288         mark~{ pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }~
1289         /SP~pdfmark ~ grestore
1290     }
1291 }
1292 \box_set_wd:Nn \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1293 \box_set_ht:Nn \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1294 \box_set_dp:Nn \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1295 \box_use_drop:N \l__pdf_backend_tmpa_box
1296 }
1297 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1298 {
1299     { pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }
1300 }
1301
1302 </dvips>
1303 <*drivers>
1304 %% all
1305 \prg_new_conditional:Npnn \__pdf_backend_xform_if_exist:n #1 { p , T , F , TF }
1306 {
1307     \int_if_exist:cTF { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1308     { \prg_return_true: }
1309     { \prg_return_false: }
1310 }
1311 \prg_new_eq_conditional:NNn \pdfxform_if_exist:n\__pdf_backend_xform_if_exist:n
1312 { TF , T , F , p }
1313 </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

```

1314 <*lua>
1315 ltx= ltx or {}
1316 ltx.__pdf      = ltx.__pdf or {}
1317 ltx.__pdf.Page = ltx.__pdf.Page or {}
1318 ltx.__pdf.Page.dflt = ltx.__pdf.Page.dflt or {}
1319 ltx.__pdf.Page.Resources = ltx.__pdf.Page.Resources or {}
1320 ltx.__pdf.Page.Resources.Properties = ltx.__pdf.Page.Resources.Properties or {}
1321 ltx.__pdf.Page.Resources.List={"ExtGState","ColorSpace","Pattern","Shading"}
1322 ltx.__pdf.object = ltx.__pdf.object or {}
1323
1324 ltx.pdf= ltx.pdf or {} -- for "public" functions
1325
1326 local __pdf = ltx.__pdf
1327 local pdf = pdf
1328
1329 local function __pdf_backend_Page_gput (name,value)
1330     __pdf.Page.dflt[name]=value
1331 end
1332

```

```

1333 local function __pdf_backend_Page_gremove (name)
1334   __pdf.Page.dflt[name]=nil
1335 end
1336
1337 local function __pdf_backend_Page_gclear ()
1338   __pdf.Page.dflt={}
1339 end
1340
1341 local function __pdf_backend_ThisPage_gput (page,name,value)
1342   __pdf.Page[page] = __pdf.Page[page] or {}
1343   __pdf.Page[page][name]=value
1344 end
1345
1346 local function __pdf_backend_ThisPage_gpush (page)
1347   local token=""
1348   local t = {}
1349   local tkeys= {}
1350   for name,value in pairs(__pdf.Page.dflt) do
1351     t[name]=value
1352   end
1353   if __pdf.Page[page] then
1354     for name,value in pairs(__pdf.Page[page]) do
1355       t[name] = value
1356     end
1357   end
1358   -- sort the table to get reliable test files.
1359   for name,value in pairs(t) do
1360     table.insert(tkeys,name)
1361   end
1362   table.sort(tkeys)
1363   for _,name in ipairs(tkeys) do
1364     token = token .. "/"..name.." "..t[name]
1365   end
1366   return token
1367 end
1368
1369 function ltx.__pdf.backend_ThisPage_gput (page,name,value) -- tex.count["g_shipout_readonly"]
1370   __pdf_backend_ThisPage_gput (page,name,value)
1371 end
1372
1373 function ltx.__pdf.backend_ThisPage_gpush (page)
1374   pdf.setpageattributes(__pdf_backend_ThisPage_gpush (page))
1375 end
1376
1377 function ltx.__pdf.backend_Page_gput (name,value)
1378   __pdf_backend_Page_gput (name,value)
1379 end
1380
1381 function ltx.__pdf.backend_Page_gremove (name)
1382   __pdf_backend_Page_gremove (name)
1383 end
1384
1385 function ltx.__pdf.backend_Page_gclear ()
1386   __pdf_backend_Page_gclear ()

```

```

1387 end
1388
1389
1390 local Properties = ltx.__pdf.Page.Resources.Properties
1391 local ResourceList= ltx.__pdf.Page.Resources.List
1392 local function __pdf_backend_PageResources_gpush (page)
1393   local token=""
1394   if Properties[page] then
1395     -- we sort the table, so that the pdf test works
1396     local t = {}
1397     for name,value in pairs (Properties[page]) do
1398       table.insert (t,name)
1399     end
1400     table.sort (t)
1401     for _,name in ipairs(t) do
1402       token = token .. "/"..name.." ".. Properties[page][name]
1403     end
1404     token = "/Properties <<"..token..>>"
1405   end
1406   for i,name in ipairs(ResourceList) do
1407     if ltx.__pdf.Page.Resources[name] then
1408       token = token .. "/"..name.." "..ltx.pdf.object_ref("__pdf/Page/Resources/"..name)
1409     end
1410   end
1411   return token
1412 end
1413
1414 -- the function is public, as I probably need it in tagpdf too ...
1415 function ltx.pdf.Page_Resources_Properties_gput (page,name,value) -- tex.count["g_shipout_re
1416   Properties[page] = Properties[page] or {}
1417   Properties[page][name]=value
1418   pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
1419 end
1420
1421 function ltx.pdf.Page_Resources_gpush(page)
1422   pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
1423 end
1424
1425 function ltx.pdf.object_ref (objname)
1426   if ltx.__pdf.object[objname] then
1427     local ref= ltx.__pdf.object[objname]
1428     return ref
1429   else
1430     return "false"
1431   end
1432 end
1433 </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	D
bool commands:	dim commands:
\bool_if:NTF . . . . . 601, 634, 658, 692, 714, 753, 801	\dim_to_decimal:n . . . . . 1240, 1242, 1243
\bool_new:N . . . . . 488	\c_zero_dim . . . . . 1129, 1130, 1131, 1270, 1271, 1272
\bool_set_true:N 942, 1024, 1117, 1216	\directlua . . . . . 59
box commands:	
\box_dp:N . . . . . 955, 1036, 1128, 1229, 1240	
\box_ht:N . . . . . 952, 1033, 1125, 1226, 1243	
\box_new:N . . . . . 50, 51, 1114	
\box_scale:Nnn . . . . . 1231	
\box_set_dp:Nn 1129, 1196, 1270, 1294	
\box_set_ht:Nn 1130, 1195, 1271, 1293	
\box_set_wd:Nn 1131, 1194, 1272, 1292	
\box_use:N . . . . . 1276	
\box_use_drop:N 1142, 1197, 1251, 1295	
\box_wd:N . . . . . 949, 1030, 1122, 1223, 1242	
	E
	exp commands:
	\exp_args:NNx . . . . . 822
	\exp_args:Nnx . . . . . 457, 636, 660
	\exp_args:Nxxx . . . . . 716, 740, 755, 779
	\exp_args:Nx . . . . . 209, 320, 359, 632, 642, 656, 666, 710, 749, 1173
	H
	hbox commands:
	\hbox_gset:Nn . . . . . 1115
	\hbox_set:Nn . . . . . 940, 1022, 1187, 1214, 1232, 1283
	hook commands:
	\hook_gput_code:nnn . . . . . 104, 448, 1273
	\hook_gput_next_code:nn . . . . . 1132
	\hook_gset_rule:nnnn . . . . . 442, 443
C	I
clist commands:	int commands:
\clist_const:Nn . . . . . 386	\int_compare:nNnTF . . . . . 911, 963, 1044
\clist_map_function:NN . . . . . 835	\int_const:Nn . . . . . 996, 1076, 1111, 1210
\clist_map_inline:Nn 395, 429, 445, 614	\int_gincr:N . . . . . 175, 546, 565, 631, 655, 709, 713, 748, 752, 1110, 1209
cs commands:	\int_if_exist:NTF . . . . . 1307
\cs_generate_variant:Nn 29, 30, 41, 42	\int_new:N . . . . . 54, 55, 56
\cs_gset_eq:NN . . . . . 602, 603, 693, 694, 802, 803	\int_use:N . . . . . 176, 177, 178, 182, 185, 186, 549, 557, 568, 576, 633, 638, 647, 657, 662, 671, 711, 718, 723, 725, 726, 730, 733, 734, 742, 750, 757, 762, 764, 765, 769, 772, 773, 781, 1004, 1010, 1083, 1091, 1182, 1238, 1264, 1288, 1299
\cs_if_exist:NTF . . . . . 398, 898	
\cs_new:Npn . . . . . 37, 62, 68, 217, 811, 1008, 1090, 1180, 1204, 1297	
\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, 895, 934, 1001, 1016, 1081, 1104, 1185, 1202, 1203, 1207, 1281	
\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, 900, 904, 913, 917, 924, 928	
	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

<p>\__kernel_backend_postscript:n . . . . . 1234, 1252, 1258, 1285</p> <p>\__kernel_pdf_name_from_unicode_e:n . . . . . 62, 68</p> <p>\__kernel_pdfdict_name:n 201, 202, 204, 432, 458, 616, 814, 825, 830, 943, 964, 975, 980, 985, 990, 1025, 1045, 1055, 1060, 1065, 1070, 1217</p> <p>\g__kernel_pdfmanagement_end_run_code_tl . . . . . 77, 84, 91</p> <p>\g__kernel_pdfmanagement_thispage_shipout_code_tl 100, 106</p> <p style="text-align: center;"><b>L</b></p> <p>lualua commands:</p> <p>\lualua: . . . . . 169, 255, 306, 345</p> <p style="text-align: center;"><b>M</b></p> <p>mode commands:</p> <p>\mode_leave_vertical: . . . 1134, 1275</p> <p style="text-align: center;"><b>P</b></p> <p>pdf commands:</p> <p>\pdf_object_ref:n . . . . . 816, 977, 982, 987, 992, 1057, 1062, 1067, 1072, 1148, 1155, 1162, 1170</p> <p>\pdf_object_ref_last: . . 847, 854, 861</p> <p>\pdf_object_unnamed_write:nn . . . . . 590, 683, 792, 846, 853, 860</p> <p>\pdf_object_write . . . . . 460</p> <p>pdf internal commands:</p> <p>\__pdf_backend:n . . . . . 139, 452, 461, 861, 926, 930, 1135, 1143, 1144, 1151, 1158, 1165, 1174, 1189</p> <p>\__pdf_backend_bdc:nn . . . . . 12, 485</p> <p>\__pdf_backend_bdc_contobj:nn . . . . . 588, 602, 681, 693, 790, 802</p> <p>\__pdf_backend_bdc_contstream:nn . . . . . 594, 603, 686, 694, 795, 803</p> <p>\__pdf_backend_bdcobject:n . . 12, 485</p> <p>\__pdf_backend_bdcobject:nn . . 12, 485</p> <p>\__pdf_backend_bmc:n . . . . . 12, 485</p> <p>\__pdf_backend_catalog_gput:nn . . 19</p> <p>\__pdf_backend_emc: . . . . . 12, 485</p> <p>\__pdf_backend_link_off: . . . . . 894, 900, 913, 924</p> <p>\__pdf_backend_link_on: . . . . . 895, 904, 917, 928</p> <p>\__pdf_backend_luastring:n . . . . . 125, 217, 226, 238, 239, 250, 265, 266</p> <p>\g__pdf_backend_name_int . . . . . 53, 546, 549, 557, 565, 568, 576, 631, 633, 638, 647, 655, 657, 662, 671, 709, 711, 748, 750</p>	<p>\__pdf_backend_Names_gpust:nn . . . . . 844, 851, 858, 867, 871</p> <p>\__pdf_backend_NamesEmbeddedFiles_add:nn . . . . . 873</p> <p>\g__pdf_backend_object_int . . . . . 1110, 1113, 1209, 1212, 1238</p> <p>\__pdf_backend_object_last: . . . . . 507, 577, 663, 672, 758, 782</p> <p>\__pdf_backend_object_new:nn 397, 447</p> <p>\__pdf_backend_object_ref:n 404, 463, 503, 558, 621, 639, 648, 719, 743</p> <p>\__pdf_backend_object_write:nn . . . . . 434, 448</p> <p>\__pdf_backend_Page_gput:nn . . 5, 146</p> <p>\__pdf_backend_Page_gremove:n 5, 146</p> <p>\g__pdf_backend_page_int . . . . . 53</p> <p>\__pdf_backend_Page_primitive:n . . . . . 5, 146</p> <p>\__pdf_backend_PageResources:n . . . . . 450, 470, 478</p> <p>\c__pdf_backend_PageResources_clist . . 385, 395, 429, 445, 614, 836</p> <p>\__pdf_backend_PageResources_gpust:n . . . . . 12, 485</p> <p>\__pdf_backend_PageResources_gpust_aux:n . . . . . 811, 837</p> <p>\__pdf_backend_PageResources_gput:nnn . . . . . 394</p> <p>\__pdf_backend_PageResources_obj_gpust: . . . . . 394</p> <p>\__pdf_backend_Pages_primitive:n 111</p> <p>\__pdf_backend_pdfmark:n . . . . . 499, 503, 507, 511, 515, 879</p> <p>\__pdf_backend_ref_label:nn . . . . . 31, 41, 178, 726, 765</p> <p>\__pdf_backend_ref_value:nn . . . . . 37, 42, 186, 734, 773</p> <p>\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</p> <p>\__pdf_backend_ThisPage_gpust:n . . . . . 5, 146</p> <p>\__pdf_backend_ThisPage_gput:nn . . . . . 5, 146</p> <p>\g__pdf_backend_thispage_shipout_tl . . . . . 5</p> <p>\l__pdf_backend_tmpa_box . . . . . 47, 940, 949, 952, 955, 995, 1022, 1030, 1033, 1036, 1075, 1187, 1194, 1195, 1196, 1197, 1214, 1223, 1226, 1229, 1231, 1240, 1242, 1243, 1251, 1283, 1292, 1293, 1294, 1295</p>
--	--





