

The l3backend-testphase package

Additional backend PDF features

L^AT_EX PDF management testphase bundle

The L^AT_EX Project*

Version 0.95r, released 2022-08-24

1 l3backend-testphase Implementation

```
1 <drivers>\ProvidesExplFile
2 <*dvipdfmx>
3   {l3backend-testphase-dvipdfmx.def}{2022-08-24}{}
4   {LaTeX~PDF~management~testphase~bundle~backend~support: dvipdfmx}
5 </dvipdfmx>
6 <*dvips>
7   {l3backend-testphase-dvips.def}{2022-08-24}{}
8   {LaTeX~PDF~management~testphase~bundle~backend~support: dvips}
9 </dvips>
10 <*dvisvgm>
11   {l3backend-testphase-dvisvgm.def}{2022-08-24}{}
12   {LaTeX~PDF~management~testphase~bundle~backend~support: dvisvgm}
13 </dvisvgm>
14 <*luatex>
15   {l3backend-testphase-luatex.def}{2022-08-24}{}
16   {LaTeX~PDF~management~testphase~bundle~backend~support: PDF output (LuaTeX)}
17 </luatex>
18 <*pdftex>
19   {l3backend-testphase-pdftex.def}{2022-08-24}{}
20   {LaTeX~PDF~management~testphase~bundle~backend~support: PDF output (pdfTeX)}
21 </pdftex>
22 <*xdvipdfmx>
23   {l3backend-testphase-xetex.def}{2022-08-24}{}
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 needed 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

```

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_tmpa_box`.)

```

\g__pdf_backend_resourceid_int
\g__pdf_backend_name_int
\g__pdf_backend_page_int

```

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

```

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

```

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

1.2 luacode

Load the lua code.

```

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

```

1.3 Converting unicode strings to a pdfname

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

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

1.4 Hooks

1.4.1 Add the “end run” hooks

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

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

1.4.2 Add the “shipout” hooks

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

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

```

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

```

1.5 The /Pages dictionary (pdfpagesattr)

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

```

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

```

145 $\langle /dvisvgm \rangle$

(End definition for `_pdf_backend_Pages_primitive:n`.)

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

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

```

146 % backend commands
147  $\langle *pdf_{tex} \rangle$ 
148 %the primitive
149 \cs_new_protected:Npn \_pdf_backend_Page_primitive:n #1
150 {
151     \tex_global:D \tex_pdfpageattr:D { #1 }
152 }
153 % the command to store default values.
154 % Uses a prop with pdflatex + dvi,
155 % sets a lua table with luatex
156 \cs_new_protected:Npn \_pdf_backend_Page_gput:nn #1 #2 %key,value
157 {
158     \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
159 }
160 % the command to remove a default value.
161 % Uses a prop with pdflatex + dvi,
162 % changes a lua table with luatex
163 \cs_new_protected:Npn \_pdf_backend_Page_gremove:n #1
164 {
165     \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
166 }
167 % the command used in the document.
168 % direct call of the primitive special with dvips/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     \_pdf_backend_ref_label:en { l3pdf\int_use:N\g__pdf_backend_resourceid_int }{abspage}
177     \tl_set:Nx \l__pdf_tmpa_tl
178     {
179         \_pdf_backend_ref_value:en {l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
180     }
181     \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl}
182     {
183         \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl}
184     }

```

```

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

```

```

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

```

```

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

```



```

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

```

378 <*drivers>
379 \clist_const:Nn \c__pdf_backend_PageResources_clist
380 {
381   ExtGState,
382   ColorSpace,
383   Pattern,
384   Shading,
385 }
386 </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:`

```

387 <*pdfTeX | luatex>
388 \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
389 {
390   \pdf_object_new:nn {__pdf/Page/Resources/#1} {dict}
391   \cs_if_exist:NT \tex_directlua:D
392   {
393     \tex_directlua:D
394     {
395       ltx.__pdf.object["__pdf/Page/Resources/#1"]
396       =
397       "\__pdf_backend_object_ref:n{__pdf/Page/Resources/#1}"
398     }
399   }
400 }
401 </pdfTeX | luatex>

```

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

```

402 <*luatex>
403 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
404 {
405   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
406   \tex_latelua:D{ltx.__pdf.Page.Resources.#1=true}
407   \tex_latelua:D
408   {
409     ltx.pdf.Page_Resources_gpush(tex.count["g_shipout_readonly_int"])
410   }
411 }
412 </luatex>
413 <*pdfTeX>
414 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
415 {
416   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
417 }
418 </pdfTeX>

```

code for end of document code

```

419 <*pdfTeX | luatex>
420 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush:
421 {
422   \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
423   {
424     \prop_if_empty:cF
425     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/##1 } }
426     {
427       \pdf_object_write:nx
428       { __pdf/Page/Resources/##1 }

```

```

429         { \pdfdict_use:n { g__pdf_Core/Page/Resources/##1} }
430     }
431 }
432 }
433 </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.

```

434 <*dvipdfmx | xdvipdfmx>
435 <xdvipdfmx> \hook_gset_rule:nnnn{shipout/firstpage}{l3backend-xetex}{after}{pdf}
436 <dvipdfmx> \hook_gset_rule:nnnn{shipout/firstpage}{l3backend-dvipdfmx}{after}{pdf}
437 %
438 \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
439 {
440     \pdf_object_new:nn { __pdf/Page/Resources/#1 } { dict }
441     \hook_gput_code:nnn{shipout/firstpage}{pdf}{\pdf_object_write:nn { __pdf/Page/Resources/
442 }
443 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1
444 {
445     \__pdf_backend:n {put~@resources~<<#1>>}
446 }
447 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
448 {
449     % this is not used for output, but there is a test if the resource is empty
450     \exp_args:Nnx
451     \prop_gput:cnn { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/#1 } }
452     { \str_convert_pdfname:n {#2} }{ #3 }
453     %objects are not filled with \pdf_object_write as this is not additive!
454     \__pdf_backend:x
455     {
456         put~\__pdf_backend_object_ref:n {__pdf/Page/Resources/#1}<</#2-#3>>
457     }
458 }
459
460 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
461 </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.

```

462 <*dvips>
463 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1 {}
464 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
465 { %only for the show command TEST!!
466     \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
467 }
468 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
469 </dvips>

```

dvipsvgm unneeded, or no-op

```

470 <*dvisvgm>
471 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1 {}
472 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3

```

```

473 { %only for the show command TEST!!
474   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
475 }
476 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
477 </dvisvgm>

```

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

1.7.1 Page resources /Properties + BDC operators

```

\__pdf_backend_bdc:nn \__pdf_backend_bdcobject:nn, \__pdf_backend_bdcobject:n,
\__pdf_backend_bdcobject:nn \__pdf_backend_bmc:n and \__pdf_backend_emc: are the backend command that cre-
\__pdf_backend_bdcobject:n ate the bdc/emc marker and store the properties. \__pdf_backend_PageResources_-
\__pdf_backend_bmc:n gpush:n outputs the /Properties and/or the other resources for the current page.
\__pdf_backend_emc:
\__pdf_backend_PageResources_gpush:n
478 % pdftex and luatex (and perhaps dvips ...) need to know if there are in a
479 % xform stream ...
480 <*drivers>
481 \bool_new:N \l__pdf_backend_xform_bool
482 </drivers>
483 <*dvips>
484 % dvips is easy: create an object, and reference it in the bdc
485 % ghostscript will then automatically replace it by a name
486 % and add the name to the /Properties dict
487 % special variant von accsupp
488 % https://chat.stackexchange.com/transcript/message/50831812#50831812
489 %
490 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2 % #1 eg. Span, #2: dict_content
491 {
492   \__pdf_backend_pdfmark:x{/#1~<<#2>>~/BDC}
493 }
494 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
495 {
496   \__pdf_backend_pdfmark:x{/#1~\__pdf_backend_object_ref:n{#2}~/BDC}
497 }
498 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span,
499 {
500   \__pdf_backend_pdfmark:x{/#1~\__pdf_backend_object_last:~/BDC}
501 }
502 \cs_set_protected:Npn \__pdf_backend_emc:
503 {
504   \__pdf_backend_pdfmark:n{/EMC} %
505 }
506 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
507 {
508   \__pdf_backend_pdfmark:n{/#1~/BMC} %
509 }
510 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
511
512 </dvips>
513 <*dvisvgm>
514 % dvisvgm should do nothing
515 %
516 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2 % #1 eg. Span, #2: dict_content

```

```

517 {}
518 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
519 {}
520 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span,
521 {}
522 \cs_set_protected:Npn \__pdf_backend_emc:
523 {}
524 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
525 {}
526 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
527
528 </dvisvgm>
529
530 % xetex has to create the entries in the /Properties manually
531 % (like the other backends)
532 % use pdfbase special
533 % https://chat.stackexchange.com/transcript/message/50832016#50832016
534 % the property is added to xform resources automatically,
535 % no need to worry about it.
536 <*dvipdfmx | xdvipdfmx>
537 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
538 {
539   \int_gincr:N \g__pdf_backend_name_int
540   \__kernel_backend_literal:x
541   {
542     pdf:code~/#1/l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC
543   }
544   \__kernel_backend_literal:x
545   {
546     pdf:put~@resources~
547     <<
548       /Properties~
549       <<
550         /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl
551         \__pdf_backend_object_ref:n { #2 }
552       >>
553     >>
554   }
555 }
556 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span
557 {
558   \int_gincr:N \g__pdf_backend_name_int
559   \__kernel_backend_literal:x
560   {
561     pdf:code~/#1/l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC
562   }
563   \__kernel_backend_literal:x
564   {
565     pdf:put~@resources~
566     <<
567       /Properties~
568       <<
569         /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl
570         \__pdf_backend_object_last:

```

```

571         >>
572     >>
573 }
574 }
575 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
576 {
577     \__kernel_backend_literal:n {pdf:code~/#1~BMC} %pdfbase
578 }
579
580 %this require management
581 \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
582 {
583     \pdf_object_unnamed_write:nn { dict }{ #2 }
584     \__pdf_backend_bdcobject:n { #1 }
585 }
586
587 \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
588 {
589     \__kernel_backend_literal:n {pdf:code~ /#1~<<#2>>~BDC }
590 }
591
592 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2
593 {
594     \bool_if:NTF \g__pdfmanagement_active_bool
595     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contobj:nn}
596     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn}
597     \__pdf_backend_bdc:nn {#1}{#2}
598 }
599 \cs_set_protected:Npn \__pdf_backend_emc:
600 {
601     \__kernel_backend_literal:n {pdf:code~EMC} %pdfbase
602 }
603 % properties are handled automatically, but the other resources should be added
604 % at shipout
605 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1
606 {
607     \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
608     {
609         \prop_if_empty:cF { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/##1} }
610         {
611             \__kernel_backend_literal:x
612             {
613                 pdf:put~@resources~
614                 <</##1~\__pdf_backend_object_ref:n {__pdf/Page/Resources/##1}>>
615             }
616         }
617     }
618 }
619 </dvipdfmx | xdvipdfmx>
620 % luatex + pdftex
621 < *luatex>
622 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
623 {
624     \int_gincr:N \g__pdf_backend_name_int

```

```

625 \exp_args:Nx\__kernel_backend_literal_page:n
626 { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
627 \bool_if:NTF \l__pdf_backend_xform_bool
628 {
629   \exp_args:Nnx\pdfdict_gput:nnn
630   { g__pdf_Core/Xform/Resources/Properties }
631   { l3pdf\int_use:N\g__pdf_backend_name_int }
632   { \__pdf_backend_object_ref:n { #2 } }
633 }
634 {
635   \exp_args:Nx \tex_latelua:D
636   {
637     ltx.pdf.Page_Resources_Properties_gput
638     (
639       tex.count["g_shipout_readonly_int"],
640       "l3pdf\int_use:N\g__pdf_backend_name_int",
641       "\__pdf_backend_object_ref:n { #2 }"
642     )
643   }
644 }
645 }
646 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1% #1 eg. Span
647 {
648   \int_gincr:N \g__pdf_backend_name_int
649   \exp_args:Nx\__kernel_backend_literal_page:n
650   { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
651   \bool_if:NTF \l__pdf_backend_xform_bool
652   {
653     \exp_args:Nnx\pdfdict_gput:nnn %no handler needed
654     { g__pdf_Core/Xform/Resources/Properties }
655     { l3pdf\int_use:N\g__pdf_backend_name_int }
656     { \__pdf_backend_object_last: }
657   }
658   {
659     \exp_args:Nx \tex_latelua:D
660     {
661       ltx.pdf.Page_Resources_Properties_gput
662       (
663         tex.count["g_shipout_readonly_int"],
664         "l3pdf\int_use:N\g__pdf_backend_name_int",
665         "\__pdf_backend_object_last:"
666       )
667     }
668   }
669 }
670 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
671 {
672   \__kernel_backend_literal_page:n { /#1~BMC }
673 }
674 \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
675 {
676   \pdf_object_unnamed_write:nn { dict } { #2 }
677   \__pdf_backend_bdcobject:n { #1 }
678 }

```

```

679 \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
680 {
681   \__kernel_backend_literal_page:n { /#1~<<#2>>~BDC }
682 }
683 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2
684 {
685   \bool_if:NTF \g__pdfmanagement_active_bool
686     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contobj:nn}
687     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn}
688     \__pdf_backend_bdc:nn {#1}{#2}
689 }
690 \cs_set_protected:Npn \__pdf_backend_emc:
691 {
692   \__kernel_backend_literal_page:n { EMC }
693 }
694
695 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
696 </luatex>
697 <*pdfTeX>
698 % pdfLaTeX is the most complicated as it has to go through the aux ...
699 % the push command is extended to take other resources too
700 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
701 {
702   \int_gincr:N \g__pdf_backend_name_int
703   \exp_args:Nx\__kernel_backend_literal_page:n
704     { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
705   % code to set the property ...
706   \int_gincr:N \g__pdf_backend_resourceid_int
707   \bool_if:NTF \l__pdf_backend_xform_bool
708   {
709     \exp_args:Nxxx\pdfdict_gput:nnn %no handler needed
710       { g__pdf_Core/Xform/Resources/Properties }
711       { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
712       { \__pdf_backend_object_ref:n { #2 } }
713   }
714   {
715     \__pdf_backend_ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
716     \tl_set:Nx \l__pdf_tmpa_tl
717       {
718         \__pdf_backend_ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
719       }
720     \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
721     {
722       \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
723     }
724     \exp_args:Nxxx\pdfdict_gput:nnn
725       { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
726       { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
727       { \__pdf_backend_object_ref:n{#2} }
728   }
729 }
730 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1% #1 eg. Span
731 {
732   \int_gincr:N \g__pdf_backend_name_int

```



```

733 \exp_args:Nx\__kernel_backend_literal_page:n
734 { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
735 % code to set the property ....
736 \int_gincr:N\g__pdf_backend_resourceid_int
737 \bool_if:NTF \l__pdf_backend_xform_bool
738 {
739   \exp_args:Nxxx\pdfdict_gput:nnn
740   { g__pdf_Core/Xform/Resources/Properties }
741   { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
742   { \__pdf_backend_object_last: }
743 }
744 {
745   \__pdf_backend_ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
746   \tl_set:Nx \l__pdf_tmpa_tl
747   {
748     \__pdf_backend_ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
749   }
750   \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties
751   {
752     \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
753   }
754   \exp_args:Nxxx\pdfdict_gput:nnn
755   { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
756   { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
757   { \__pdf_backend_object_last: }
758   %\pdfdict_show:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
759 }
760 }
761 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
762 {
763   \__kernel_backend_literal_page:n { /#1~BMC }
764 }
765 \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
766 {
767   \pdf_object_unnamed_write:nn { dict } { #2 }
768   \__pdf_backend_bdcobject:n { #1 }
769 }
770 \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
771 {
772   \__kernel_backend_literal_page:n { /#1~<<#2>>~BDC }
773 }
774 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2
775 {
776   \bool_if:NTF \g__pdfmanagement_active_bool
777   {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contobj:nn}
778   {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn}
779   \__pdf_backend_bdc:nn {#1}{#2}
780 }
781 \cs_set_protected:Npn \__pdf_backend_emc:
782 {
783   \__kernel_backend_literal_page:n { EMC }
784 }
785
786 \cs_new:Npn \__pdf_backend_PageResources_gpush_aux:n #1 %#1 ExtGState etc

```

```

787 {
788   \prop_if_empty:cF
789   { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/#1} }
790   {
791     \pdffdict_item:ne { #1 }{\pdf_object_ref:n {__pdf/Page/Resources/#1}}
792   }
793 }
794
795 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1
796 {
797   \exp_args:NNx \tex_global:D \tex_pdfpageresources:D
798   {
799     \prop_if_exist:cT
800     { \__kernel_pdffdict_name:n { g__pdf_Core/backend_Page#1/Resources/Properties } }
801     {
802       /Properties~
803       <<
804       \prop_map_function:cN
805       { \__kernel_pdffdict_name:n { g__pdf_Core/backend_Page#1/Resources/Property
806       \pdffdict_item:ne
807       >>
808     }
809     %% add ExtGState etc
810     \clist_map_function:NN
811     \c__pdf_backend_PageResources_clist
812     \__pdf_backend_PageResources_gpush_aux:n
813   }
814 }
815
816 \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.

```

817 % pdfLATEX
818 \pdfTeX
819 \cs_new_protected:Npn \__pdf_backend_Names_gpush:nn #1 #2 % #1 name of name tree, #2 array co
820 {
821   \pdf_object_unnamed_write:nn {dict} {/Names [#2] }
822   \tex_pdfnames:D {/#1~\pdf_object_ref_last:}
823 }
824 \pdfTeX
825 \luatex
826 \cs_new_protected:Npn \__pdf_backend_Names_gpush:nn #1 #2 % #1 name of name tree, #2 array co

```

```

827 {
828   \pdf_object_unnamed_write:nn {dict} {/Names [#2] }
829   \tex_pdfextension:D~names~ {/#1~\pdf_object_ref_last:}
830 }
831 </luatex>
832 <*dvipdfmx | xdvipdfmx>
833 \cs_new_protected:Npn \__pdf_backend_Names_gpush:nn #1 #2 % #1 name of name tree, #2 array co
834 {
835   \pdf_object_unnamed_write:nn {dict} {/Names [#2] }
836   \__pdf_backend:x {put~@names~<</#1~\pdf_object_ref_last: >>}
837 }
838 </dvipdfmx | xdvipdfmx>
839
840 %dvips: noop
841 <*dvips>
842 \cs_new_protected:Npn \__pdf_backend_Names_gpush:nn #1 #2 {}
843 </dvips>
844 %dvisvgm: noop
845 <*dvisvgm>
846 \cs_new_protected:Npn \__pdf_backend_Names_gpush:nn #1 #2 {}
847 </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.

```

848 <*pdftex | luatex | dvipdfmx | xdvipdfmx>
849 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:nn #1 #2 {}
850 </pdftex | luatex | dvipdfmx | xdvipdfmx>
851 <*dvips>
852 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:nn #1 #2
853 {
854   \__pdf_backend_pdfmark:x
855   {
856     /Name~#1~
857     /FS~#2~
858     /EMBED
859   }
860 }
861 </dvips>
862 <*dvisvgm>
863 %no op. Or is there any sensible use for it?
864 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:nn #1 #2
865 {}
866
867 </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.

```

868 <*drivers>
869 \cs_new_protected:Npn \__pdf_backend_link_off: {}
870 \cs_new_protected:Npn \__pdf_backend_link_on: {}
871 </drivers>
872 <*pdftex>
873 \cs_if_exist:NT \pdfrunninglinkoff
874 {
875   \cs_set_protected:Npn \__pdf_backend_link_off:
876   {
877     \pdfrunninglinkoff
878   }
879   \cs_set_protected:Npn \__pdf_backend_link_on:
880   {
881     \pdfrunninglinkon
882   }
883 }
884 </pdftex>
885 <*luatex>
886 \int_compare:nNnT {\tex_luatexversion:D } > {112}
887 {
888   \cs_set_protected:Npn \__pdf_backend_link_off:
889   {
890     \pdfextension linkstate 1
891   }
892   \cs_set_protected:Npn \__pdf_backend_link_on:
893   {
894     \pdfextension linkstate 0
895   }
896 }
897 </luatex>
898 <*dviPDFmx | xdvipdfmx>
899 \cs_set_protected:Npn \__pdf_backend_link_off:
900 {
901   \__pdf_backend:n { nolink }
902 }
903 \cs_set_protected:Npn \__pdf_backend_link_on:
904 {
905   \__pdf_backend:n { link }
906 }
907 </dviPDFmx | xdvipdfmx>

```

1.8.3 Form XObject / 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      908 <*pdftex>
\__pdf_backend_xform_ref:n      909 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
                               910 % #1 name
                               911 % #2 attributes
                               912 % #3 resources

```

```

913 % #4 content, not necessarily a box!
914 {
915   \hbox_set:Nn \l__pdf_backend_tmpa_box
916   {
917     \bool_set_true:N \l__pdf_backend_xform_bool
918     \prop_gclear:c {\__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties }}
919     #4
920   }
921   %store the dimensions
922   \tl_const:cx
923   { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
924   { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
925   \tl_const:cx
926   { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
927   { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
928   \tl_const:cx
929   { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
930   { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
931   %% do we need to test if #2 and #3 are empty??
932   \tex_immediate:D \tex_pdfxform:D
933   ~ attr ~ { #2 }
934   %% which other resources should be default? Is an argument actually needed?
935   ~ resources ~
936   {
937     #3
938     \int_compare:nNnT
939     { \prop_count:c { \__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties } }
940     >
941     { 0 }
942     {
943       /Properties~
944       <<
945       \pdffdict_use:n { g__pdf_Core/Xform/Resources/Properties }
946       >>
947     }
948
949     \prop_if_empty:cF
950     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
951     {
952       /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
953     }
954     \prop_if_empty:cF
955     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
956     {
957       /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
958     }
959     \prop_if_empty:cF
960     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Shading } }
961     {
962       /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
963     }
964     \prop_if_empty:cF
965     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
966     {

```

```

967         /ColorSpace~ \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
968     }
969 }
970 \l__pdf_backend_tmpa_box
971 \int_const:cn
972 { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
973 { \tex_pdflastxform:D }
974 }
975
976 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
977 {
978     \tex_pdfrefxform:D
979     \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
980     \scan_stop:
981 }
982
983 \cs_new:Npn \__pdf_backend_xform_ref:n #1
984 {
985     \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R
986 }
987 </pdftex>
988 <*luatex>
989 %luatex
990 %nearly identical but not completely ...
991 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
992 % #1 name
993 % #2 attributes
994 % #3 resources
995 % #4 content, not necessarily a box!
996 {
997     \hbox_set:Nn \l__pdf_backend_tmpa_box
998     {
999         \bool_set_true:N \l__pdf_backend_xform_bool
1000         \prop_gclear:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties }
1001             #4
1002         }
1003         \tl_const:cx
1004         { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1005         { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1006         \tl_const:cx
1007         { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1008         { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1009         \tl_const:cx
1010         { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1011         { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1012         %% do we need to test if #2 and #3 are empty??
1013         \tex_immediate:D \tex_pdfxform:D
1014         ~ attr ~ { #2 }
1015         %% which resources should be default? Is an argument actually needed?
1016         ~ resources ~
1017         {
1018             #3
1019             \int_compare:nNnT
1020                 {\prop_count:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties

```

```

1021     >
1022     { 0 }
1023     {
1024         /Properties~
1025         <<
1026         \pdfdict_use:n { g__pdf_Core/Xform/Resources/Properties }
1027         >>
1028     }
1029     \prop_if_empty:cF
1030     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
1031     {
1032         /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1033     }
1034     \prop_if_empty:cF
1035     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
1036     {
1037         /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1038     }
1039     \prop_if_empty:cF
1040     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Shading } }
1041     {
1042         /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1043     }
1044     \prop_if_empty:cF
1045     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
1046     {
1047         /ColorSpace~ \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1048     }
1049 }
1050 \l__pdf_backend_tmpa_box
1051 \int_const:cn
1052 { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1053 { \tex_pdflastxform:D }
1054 }
1055
1056 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 %protected as with xelatex
1057 {
1058     \tex_pdfrefxform:D \int_use:c
1059     {
1060         c__pdf_backend_xform_ \tl_to_str:n {#1} _int
1061     }
1062     \scan_stop:
1063 }
1064
1065 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1066 { \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R }
1067
1068 </luatex>
1069 <*dvipdfmx | xdvipdfmx>
1070 % xetex
1071 % it needs a bit testing if it really works to set the box to 0 before the special ...
1072 % does it disturb viewing the xobject?
1073 % what happens with the resources (bdc)? (should work as they are specials too)
1074 % xetex requires that the special is in horizontal mode. This means it affects

```

```

1075 % typesetting. But we can no delay the whole form code to shipout
1076 % as the object reference and the size is often wanted on the current page.
1077 % so we need to allocate a box - but probably they won't be thousands xform
1078 % in a document so it shouldn't matter.
1079 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
1080 % #1 name
1081 % #2 attributes
1082 % #3 resources
1083 % #4 content, not necessarily a box!
1084 {
1085     \int_gincr:N \g__pdf_backend_object_int
1086     \int_const:cn
1087     { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1088     { \g__pdf_backend_object_int }
1089     \box_new:c { g__pdf_backend_xform_#1_box }
1090     \hbox_gset:cn { g__pdf_backend_xform_#1_box }
1091     {
1092         \bool_set_true:N \l__pdf_backend_xform_bool
1093         #4
1094     }
1095     \tl_const:cx
1096     { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1097     { \tex_the:D \box_wd:c { g__pdf_backend_xform_#1_box } }
1098     \tl_const:cx
1099     { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1100     { \tex_the:D \box_ht:c { g__pdf_backend_xform_#1_box } }
1101     \tl_const:cx
1102     { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1103     { \tex_the:D \box_dp:c { g__pdf_backend_xform_#1_box } }
1104     \box_set_dp:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1105     \box_set_ht:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1106     \box_set_wd:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1107     \hook_gput_next_code:nn {shipout/background}
1108     {
1109         \mode_leave_vertical: %needed, the xform disappears without it.
1110         \__pdf_backend:x
1111         {
1112             bobj ~ \__pdf_backend_xform_ref:n { #1 }
1113             \c_space_tl width ~ \pdfxform_wd:n { #1 }
1114             \c_space_tl height ~ \pdfxform_ht:n { #1 }
1115             \c_space_tl depth ~ \pdfxform_dp:n { #1 }
1116         }
1117         \box_use_drop:c { g__pdf_backend_xform_#1_box }
1118         \__pdf_backend:x {put ~ @resources ~<<#3>> }
1119         \__pdf_backend:x
1120         {
1121             put~ @resources ~
1122             <<
1123             /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1124             >>
1125         }
1126         \__pdf_backend:x
1127         {
1128             put~ @resources ~

```



```

1129         <<
1130         /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1131         >>
1132     }
1133     \__pdf_backend:x
1134     {
1135         put~ @resources ~
1136         <<
1137         /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1138         >>
1139     }
1140     \__pdf_backend:x
1141     {
1142         put~ @resources ~
1143         <<
1144         /ColorSpace~
1145         \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1146         >>
1147     }
1148     \exp_args:Nx
1149     \__pdf_backend:x {exobj ~<<#2>>}
1150 }
1151 }
1152
1153
1154
1155 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1156 {
1157     @pdf.xform \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1158 }
1159
1160 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1161 {
1162     \hbox_set:Nn \l__pdf_backend_tmpa_box
1163     {
1164         \__pdf_backend:x
1165         {
1166             uxobj~ \__pdf_backend_xform_ref:n { #1 }
1167         }
1168     }
1169     \box_set_wd:Nn \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1170     \box_set_ht:Nn \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1171     \box_set_dp:Nn \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1172     \box_use_drop:N \l__pdf_backend_tmpa_box
1173 }
1174 </dvipdfmx | xdvipdfmx>
1175 <*dvisvgm>
1176 % unclear what it should do!!
1177 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4 {}
1178 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 {}
1179 \cs_new:Npn \__pdf_backend_xform_ref:n {}
1180 </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. We need some temporary variables to store dimensions

```

1181 <*dvips>
1182 \tl_new:N \l__pdf_backend_xform_tmpwd_tl
1183 \tl_new:N \l__pdf_backend_xform_tmpdp_tl
1184 \tl_new:N \l__pdf_backend_xform_tmph_t_tl

1185 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4 % #1 name, #2 attribute, #4
1186 {
1187   \int_gincr:N \g__pdf_backend_object_int
1188   \int_const:cn
1189     { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1190     { \g__pdf_backend_object_int }
1191
1192   \hbox_set:Nn \l__pdf_backend_tmpa_box
1193     {
1194       \bool_set_true:N \l__pdf_backend_xform_bool
1195       \prop_gc_clear:c {\__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties }}
1196       #4
1197     }
1198   %store the dimensions
1199   \tl_const:cx
1200     { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1201     { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1202   \tl_const:cx
1203     { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1204     { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1205   \tl_const:cx
1206     { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1207     { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1208   %store content dimensions in DPI units (Dots) (code from issue 25)
1209   \tl_set:Nx \l__pdf_backend_xform_tmpwd_tl
1210     {
1211       \dim_to_decimal_in_sp:n{ \box_wd:N \l__pdf_backend_tmpa_box }~
1212       65536~div~72.27~div~DVImag~mul~Resolution~mul~
1213     }
1214   \tl_set:Nx \l__pdf_backend_xform_tmph_t_tl
1215     {
1216       \dim_to_decimal_in_sp:n{ \box_ht:N \l__pdf_backend_tmpa_box }~
1217       65536~div~72.27~div~DVImag~mul~VResolution~mul~
1218     }
1219   \tl_set:Nx \l__pdf_backend_xform_tmpdp_tl
1220     {
1221       \dim_to_decimal_in_sp:n{ \box_dp:N \l__pdf_backend_tmpa_box }~
1222       65536~div~72.27~div~DVImag~mul~VResolution~mul~
1223     }
1224   % mirror the box
1225   %\box_scale:Nnn \l__pdf_backend_tmpa_box {1} {-1}
1226   \hbox_set:Nn \l__pdf_backend_tmpb_box
1227     {
1228       \__kernel_backend_postscript:x
1229       {
1230         gsave~currentpoint~
1231         initclip~ % restore default clipping path (page device/whole page)

```

```

1232      clippath~pathbbox~newpath~pop~pop~
1233      \tl_use:N\l__pdf_backend_xform_tmpdp_tl~add~translate~
1234      mark~
1235      /_objdef~{ pdf.obj \int_use:N\g__pdf_backend_object_int }\c_space_tl~
1236      /BBox[
1237        0~
1238        \tl_use:N\l__pdf_backend_xform_tmpt_tl~
1239        \tl_use:N\l__pdf_backend_xform_tmpwd_tl~
1240        \tl_use:N\l__pdf_backend_xform_tmpdp_tl~
1241        neg
1242      ]
1243      \str_if_eq:eeF{#1}{}
1244      {
1245        product~(Distiller)~search~{pop~pop~pop~#2}{pop}ifelse~
1246      }
1247      /BP~pdfmark~1~-1~scale~neg~exch~neg~exch~translate
1248    }
1249    \box_use_drop:N\l__pdf_backend_tmpa_box
1250    \__kernel_backend_postscript:n
1251    {
1252      mark ~ /EP~pdfmark ~ grestore
1253    }
1254    \str_if_eq:eeF{#1}{}
1255    {
1256      \__kernel_backend_postscript:x
1257      {
1258        product~(Ghostscript)~search~
1259        {
1260          pop~pop~pop~
1261          mark~
1262          { pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }
1263          ~<<#2>>~/PUT~pdfmark
1264        }{pop}ifelse
1265      }
1266    }
1267  }
1268  \box_set_dp:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1269  \box_set_ht:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1270  \box_set_wd:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1271  \hook_gput_code:nnn {begindocument/end}{pdfxform}
1272  {
1273    \mode_leave_vertical:
1274    \box_use:N\l__pdf_backend_tmpb_box
1275  }
1276 }
1277
1278
1279 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1280 {
1281   \hbox_set:Nn \l__pdf_backend_tmpa_box
1282   {
1283     \__kernel_backend_postscript:x
1284     {
1285       gsave~currentpoint~translate~1~-1~scale~

```

```

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

```

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

1.9 Structure Destinations

Standard destinations consist of a reference to a page in the pdf and instructions how to display it—typically they will put a specific location in the left top corner of the viewer and so give the impression that a link jumped to the word in this place. But in reality they are not connected to the content.

Starting with pdf 2.0 destinations can in a tagged PDF also point to a structure, to a /StructElem object. GoTo links can then additionally to the /D key pointing to a page destination also point to such a structure destination with an /SD key. Programs that e.g. convert such a PDF to html can then create better links. (According to the reference, PDF-viewer should prefer the structure destination over the page destination, but as far as it is known this isn't done yet.)

Currently structure destinations and GoTo links making use of it could natively only be created with the dvipdfmx backend. With pdftex and lualatex it was only possible to create a restricted type which used only the “Fit” mode. Starting with T_EXlive 2022 (earlier in miktex) both engine will know new keywords which allow to create structure destination easily.

The following backend code prepares the use of structure destinations. The general idea is that if structure destinations are used, they should be used always. So we define alternative commands which can be activated by mapping them to the standard backend commands.

`\l_pdf_current_structure_destination_tl` This commands holds the name of the structure object to use in the next command which creates a destination. The code which activates structure destinations must also ensure that it has a sensible, expandable content. `tagpdf` for example will define it as

```
\tl_set:Nn \l_pdf_current_structure_destination_tl { __tag/struct/\g__tag_struct_stack
1312 <*drivers>
1313 \tl_new:N \l_pdf_current_structure_destination_tl
1314 </drivers>
```

(End definition for `\l_pdf_current_structure_destination_tl`. This function is documented on page ??.)

We will define alternatives for three backend commands:

```
\__pdf_backend_destination:nn -> \__pdf_backend_structure_destination:nn
\__pdf_backend_destination:nnnn -> \__pdf_backend_structure_destination:nnnn
\__pdf_backend_link_begin_goto:nnw -> \__pdf_backend_link_begin_structure_goto:nnw
```

Activating means mapping them onto the original commands. Be aware that not all engines and compilation routes support structure destinations, for them the command will be a no-op.

`\pdf_activate_structure_destination:`

```
1315 <*drivers>
1316 \cs_new_protected:Npn \pdf_activate_structure_destination:
1317 {
1318   \cs_gset_eq:NN \__pdf_backend_destination:nn \__pdf_backend_structure_destination:nn
1319   \cs_gset_eq:NN \__pdf_backend_destination:nnnn \__pdf_backend_structure_destination:nnnn
1320   \cs_gset_eq:NN \__pdf_backend_link_begin_goto:nnw \__pdf_backend_link_begin_structure_goto:nnw
1321 }
1322 </drivers>
```

(End definition for `\pdf_activate_structure_destination:`. This function is documented on page ??.)

Now the driver dependant parts. By default the new commands are simply copies of the original commands. We adapt them then for the engines and engine version which provide support for structure destinations.

```
1323 <*drivers>
1324 \cs_set_eq:NN \__pdf_backend_structure_destination:nn \__pdf_backend_destination:nn
1325 \cs_set_eq:NN \__pdf_backend_structure_destination:nnnn \__pdf_backend_destination:nnnn
1326 \cs_set_eq:NN \__pdf_backend_link_begin_structure_goto:nnw \__pdf_backend_link_begin_goto:nnw
1327 </drivers>
```

`__pdf_backend_structure_destination:nn` This command is the backend command to create a destination. It should in parallel create also a structure destination. At first xetex/dvipdfmx. The structure destination is an array, so we use `obj` for it so that we can reference it:

```
1328 <*xdvipdfmx|dvipdfmx>
1329 \cs_set_protected:Npn \__pdf_backend_structure_destination:nn #1#2
1330 {
1331   \__pdf_backend:x
1332   {
1333     dest ~ ( \exp_not:n {#1} )
1334     [
1335       @thispage
1336       \str_case:nnF {#2}
```

```

1337     {
1338         { xyz }    { /XYZ ~ @xpos ~ @ypos ~ null }
1339         { fit }    { /Fit }
1340         { fitb }   { /FitB }
1341         { fitbh }  { /FitBH }
1342         { fitbv }  { /FitBV ~ @xpos }
1343         { fith }   { /FitH ~ @ypos }
1344         { fitv }   { /FitV ~ @xpos }
1345         { fitr }   { /Fit }
1346     }
1347     { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1348 ]
1349 }

```

We test if the structure object exist. The object of the structure destination gets the name `@pdf.Sdest.<destname>`, where `<destname>` is the name of the standard destination so that we can reference it in the GoTo links.

```

1350 \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1351 {
1352     \__pdf_backend:x
1353     {
1354         obj ~ @pdf.SDest.\exp_not:n{#1}
1355         [
1356             \exp_args:Ne \pdf_object_ref:n { \l_pdf_current_structure_destination_tl }
1357             \str_case:nnF {#2}
1358             {
1359                 { xyz }    { /XYZ ~ @xpos ~ @ypos ~ null }
1360                 { fit }    { /Fit }
1361                 { fitb }   { /FitB }
1362                 { fitbh }  { /FitBH }
1363                 { fitbv }  { /FitBV ~ @xpos }
1364                 { fith }   { /FitH ~ @ypos }
1365                 { fitv }   { /FitV ~ @xpos }
1366                 { fitr }   { /Fit }
1367             }
1368             { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1369         ]
1370     }
1371 }
1372 }

```

The second destination command is for the boxed destination. Here we need to define an new auxiliary command:

```

1373 \cs_new_protected:Npn \__pdf_backend_structure_destination_aux:nnnn #1#2#3#4
1374 {
1375     \vbox_to_zero:n
1376     {
1377         \__kernel_kern:n {#4}
1378         \hbox:n
1379         {
1380             \__pdf_backend:n { obj ~ @pdf_ #2 _llx ~ @xpos }
1381             \__pdf_backend:n { obj ~ @pdf_ #2 _lly ~ @ypos }
1382         }
1383         \tex_vss:D
1384     }

```

```

1385 \__kernel_kern:n {#1}
1386 \vbox_to_zero:n
1387 {
1388   \__kernel_kern:n { -#3 }
1389   \hbox:n
1390   {
1391     \__pdf_backend:n
1392     {
1393       dest ~ (#2)
1394       [
1395         @thispage
1396         /FitR ~
1397         @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1398         @xpos ~ @ypos
1399       ]
1400     }

```

Here we add the structure destination to the same box

```

1401   \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1402   {
1403     \__pdf_backend:x
1404     {
1405       obj ~ @pdf.SDest.\exp_not:n{#2}
1406       [
1407         \exp_args:Ne \pdf_object_ref:n { \l_pdf_current_structure_destination_
1408         /FitR ~
1409         @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1410         @xpos ~ @ypos
1411       ]
1412     }
1413   }
1414 }
1415 \tex_vss:D
1416 }
1417 \__kernel_kern:n { -#1 }
1418 }

```

And now we redefine the destination command:

```

1419 \cs_set_protected:Npn \__pdf_backend_structure_destination:nnnn #1#2#3#4
1420 {
1421   \exp_args:Ne \__pdf_backend_structure_destination_aux:nnnn
1422   { \dim_eval:n {#2} } {#1} {#3} {#4}
1423 }

```

At last the goto link.

```

1424 \cs_set_protected:Npn \__pdf_backend_link_begin_structure_goto:nnw #1#2
1425 {
1426   \__pdf_backend_link_begin:n { #1 /Subtype /Link /A << /S /GoTo /D ( #2 ) /SD~@pdf.SDest.
1427 }
1428 </xdvipdfmx | dvipdfmx>

```

(End definition for __pdf_backend_structure_destination:nn.)

Now pdftex. We only redefine for version 1.40 revision 24 or later.

```

1429 <*pdftex>
1430 \bool_lazy_and:nnT

```

```

1431 { \int_compare_p:nNn {\tex_pdfversion:D } > {139} }
1432 { \int_compare_p:nNn {\tex_pdfrevision:D } > {23} }
1433 {
1434   \cs_set_protected:Npn \__pdf_backend_structure_destination:nn #1#2
1435   {
1436     \tex_pdfdest:D
1437       name {#1}
1438       \str_case:nnF {#2}
1439       {
1440         { xyz } { xyz }
1441         { fit } { fit }
1442         { fitb } { fitb }
1443         { fitbh } { fitbh }
1444         { fitbv } { fitbv }
1445         { fith } { fith }
1446         { fitv } { fitv }
1447         { fitr } { fitr }
1448       }
1449       { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1450       \scan_stop:
1451     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1452     {
1453       \tex_pdfdest:D
1454       struct~
1455       \int_use:c
1456       { c__pdf_backend_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structur
1457       name {#1}
1458       \str_case:nnF {#2}
1459       {
1460         { xyz } { xyz }
1461         { fit } { fit }
1462         { fitb } { fitb }
1463         { fitbh } { fitbh }
1464         { fitbv } { fitbv }
1465         { fith } { fith }
1466         { fitv } { fitv }
1467         { fitr } { fitr }
1468       }
1469       { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1470       \scan_stop:
1471     }
1472   }
1473   \cs_set_protected:Npn \__pdf_backend_destination:nnnn #1#2#3#4
1474   {
1475     \tex_pdfdest:D
1476     name {#1}
1477     fitr ~
1478     width \dim_eval:n {#2} ~
1479     height \dim_eval:n {#3} ~
1480     depth \dim_eval:n {#4} \scan_stop:
1481     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1482     {
1483       \tex_pdfdest:D
1484       struct~

```



```

1485         \int_use:c
1486         { c__pdf_backend_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structure_
1487         name {#1}
1488         fitr ~
1489         width \dim_eval:n {#2} ~
1490         height \dim_eval:n {#3} ~
1491         depth \dim_eval:n {#4} \scan_stop:
1492     }
1493 }
1494 \cs_set_protected:Npn \__pdf_backend_link_begin_structure_goto:nnw #1#2
1495 {
1496     \__pdf_backend_link_begin:nnnw {#1} { goto~struct~name~{#2}~name } {#2}
1497 }
1498 }
1499 </pdfTeX>

```

luatex is quite similar to pdfTeX. Mostly the test for the version is different

```

1500 <*luatex>
1501 \int_compare:nNtT {\directlua{tex.print(status.list()["development_id"])} } > {7468}
1502 {
1503     \cs_set_protected:Npn \__pdf_backend_structure_destination:nn #1#2
1504     {
1505         \tex_pdfextension:D dest
1506         name {#1}
1507         \str_case:nnF {#2}
1508         {
1509             { xyz } { xyz }
1510             { fit } { fit }
1511             { fitb } { fitb }
1512             { fitbh } { fitbh }
1513             { fitbv } { fitbv }
1514             { fith } { fith }
1515             { fitv } { fitv }
1516             { fitr } { fitr }
1517         }
1518         { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1519         \scan_stop:
1520     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1521     {
1522         \tex_pdfextension:D dest
1523         struct~
1524         \int_use:c
1525         { c__pdf_backend_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structur
1526         name {#1}
1527         \str_case:nnF {#2}
1528         {
1529             { xyz } { xyz }
1530             { fit } { fit }
1531             { fitb } { fitb }
1532             { fitbh } { fitbh }
1533             { fitbv } { fitbv }
1534             { fith } { fith }
1535             { fitv } { fitv }
1536             { fitr } { fitr }
1537         }

```

```

1538         { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1539         \scan_stop:
1540     }
1541 }
1542 \cs_set_protected:Npn \__pdf_backend_destination:nnnn #1#2#3#4
1543 {
1544     \tex_pdfextension:D dest
1545     name {#1}
1546     fitr ~
1547     width \dim_eval:n {#2} ~
1548     height \dim_eval:n {#3} ~
1549     depth \dim_eval:n {#4} \scan_stop:
1550     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1551     {
1552         \tex_pdfextension:D dest
1553         struct~
1554         \int_use:c
1555         { c__pdf_backend_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structure_
1556         name {#1}
1557         fitr ~
1558         width \dim_eval:n {#2} ~
1559         height \dim_eval:n {#3} ~
1560         depth \dim_eval:n {#4} \scan_stop:
1561     }
1562 }
1563 \cs_set_protected:Npn \__pdf_backend_link_begin_structure_goto:nnw #1#2
1564 {
1565     \__pdf_backend_link_begin:nnnw {#1} { goto~struct~name~{#2}~name } {#2}
1566 }
1567 }
1568 \</luatex>

```

1.10 Settings for regression tests

When doing pdf based regression tests some meta data in the pdf should have fixed values to get identical pdf's. We define here the backend dependant part. The main command is then in l3pdfmeta

```

1569 <*drivers>
1570 \cs_new_protected:Npn \__pdf_backend_set_regression_data:
1571 {
1572     \sys_gset_rand_seed:n{1000}
1573     \pdfmanagement_add:nnn{Info}{Creator}{(TeX)}
1574 </drivers>
1575 <*dvips>
1576     \__kernel_backend_literal:e{!~<</DocumentUUID~(DocumentUUID)>>~setpagedevice}
1577     \__kernel_backend_literal:e{!~<</InstanceUUID~(InstanceUUID)>>~setpagedevice}
1578 </dvips>
1579 <*dviPDFmx>
1580     \pdfmanagement_add:nnn{Info}{Producer}{(dviPDFmx)}
1581     \__kernel_backend_literal:e
1582     {pdf:trailerid [~
1583     <00112233445566778899aabbccddeeff>~
1584     <00112233445566778899aabbccddeeff>~

```

```

1585     ]}
1586 </dvipdfmx>
1587 <*xdvipdfmx>
1588     \pdfmanagement_add:nnn{Info}{Producer}{(xetex)}
1589     \__kernel_backend_literal:e
1590     {pdf:trailerid [~
1591       <00112233445566778899aabbccddeeff>~
1592       <00112233445566778899aabbccddeeff>~
1593     ]}
1594 </xdvipdfmx>
1595 <*pdftex>
1596     \pdfmanagement_add:nnn{Info}{Producer}{(pdfTeX)}
1597     \tex_pdfsuppressptexinfo:D 7 \scan_stop:
1598     \pdftrailerid{2350CAD05F8A7AF0AA4058486855344F}
1599 </pdftex>
1600 <*luatex>
1601     \pdfmanagement_add:nnn{Info}{Producer}{(LuaTeX)}
1602     \tex_pdfvariable:D suppressoptionalinfo 7\relax
1603     \tex_pdfvariable:D trailerid
1604     {[~
1605       <2350CAD05F8A7AF0AA4058486855344F>~
1606       <2350CAD05F8A7AF0AA4058486855344F>~
1607     ]}
1608 </luatex>
1609 <*drivers>
1610     \pdfmanagement_add:nnn{Info}{CreationDate}{(D:20010101205959-00'00')}
1611     \pdfmanagement_add:nnn{Info}{ModDate}{(D:20010101205959-00'00')}
1612   }
1613 </drivers>

```

1.11 lua code for luatex

```

1614 <*lua>
1615 ltx= ltx or {}
1616 ltx.__pdf      = ltx.__pdf or {}
1617 ltx.__pdf.Page = ltx.__pdf.Page or {}
1618 ltx.__pdf.Page.dflt = ltx.__pdf.Page.dflt or {}
1619 ltx.__pdf.Page.Resources = ltx.__pdf.Resources or {}
1620 ltx.__pdf.Page.Resources.Properties = ltx.__pdf.Page.Resources.Properties or {}
1621 ltx.__pdf.Page.Resources.List={"ExtGState","ColorSpace","Pattern","Shading"}
1622 ltx.__pdf.object = ltx.__pdf.object or {}
1623
1624 ltx.pdf= ltx.pdf or {} -- for "public" functions
1625
1626 local __pdf = ltx.__pdf
1627 local pdf = pdf
1628
1629 local function __pdf_backend_Page_gput (name,value)
1630   __pdf.Page.dflt[name]=value
1631 end
1632
1633 local function __pdf_backend_Page_gremove (name)
1634   __pdf.Page.dflt[name]=nil
1635 end

```

```

1636
1637 local function __pdf_backend_Page_gclear ()
1638   __pdf.Page.dflt={}
1639 end
1640
1641 local function __pdf_backend_ThisPage_gput (page,name,value)
1642   __pdf.Page[page] = __pdf.Page[page] or {}
1643   __pdf.Page[page][name]=value
1644 end
1645
1646 local function __pdf_backend_ThisPage_gpush (page)
1647   local token=""
1648   local t = {}
1649   local tkeys= {}
1650   for name,value in pairs(__pdf.Page.dflt) do
1651     t[name]=value
1652   end
1653   if __pdf.Page[page] then
1654     for name,value in pairs(__pdf.Page[page]) do
1655       t[name] = value
1656     end
1657   end
1658   -- sort the table to get reliable test files.
1659   for name,value in pairs(t) do
1660     table.insert(tkeys,name)
1661   end
1662   table.sort(tkeys)
1663   for _,name in ipairs(tkeys) do
1664     token = token .. "/"..name.." " ..t[name]
1665   end
1666   return token
1667 end
1668
1669 function ltx.__pdf.backend_ThisPage_gput (page,name,value) -- tex.count["g_shipout_readonly"]
1670   __pdf_backend_ThisPage_gput (page,name,value)
1671 end
1672
1673 function ltx.__pdf.backend_ThisPage_gpush (page)
1674   pdf.setpageattributes(__pdf_backend_ThisPage_gpush (page))
1675 end
1676
1677 function ltx.__pdf.backend_Page_gput (name,value)
1678   __pdf_backend_Page_gput (name,value)
1679 end
1680
1681 function ltx.__pdf.backend_Page_gremove (name)
1682   __pdf_backend_Page_gremove (name)
1683 end
1684
1685 function ltx.__pdf.backend_Page_gclear ()
1686   __pdf_backend_Page_gclear ()
1687 end
1688
1689

```

```

1690 local Properties = ltx.__pdf.Page.Resources.Properties
1691 local ResourceList= ltx.__pdf.Page.Resources.List
1692 local function __pdf_backend_PageResources_gpush (page)
1693   local token=""
1694   if Properties[page] then
1695     -- we sort the table, so that the pdf test works
1696     local t = {}
1697     for name,value in pairs (Properties[page]) do
1698       table.insert (t,name)
1699     end
1700     table.sort (t)
1701     for _,name in ipairs(t) do
1702       token = token .. "/"..name.." ".. Properties[page][name]
1703     end
1704     token = "/Properties <<"..token..">>"
1705   end
1706   for i,name in ipairs(ResourceList) do
1707     if ltx.__pdf.Page.Resources[name] then
1708       token = token .. "/"..name.." "..ltx.pdf.object_ref("__pdf/Page/Resources/"..name)
1709     end
1710   end
1711   return token
1712 end
1713
1714 -- the function is public, as I probably need it in tagpdf too ...
1715 function ltx.pdf.Page_Resources_Properties_gput (page,name,value) -- tex.count["g_shipout_re
1716   Properties[page] = Properties[page] or {}
1717   Properties[page][name]=value
1718   pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
1719 end
1720
1721 function ltx.pdf.Page_Resources_gpush(page)
1722   pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
1723 end
1724
1725 function ltx.pdf.object_ref (objname)
1726   if ltx.__pdf.object[objname] then
1727     local ref= ltx.__pdf.object[objname]
1728     return ref
1729   else
1730     return "false"
1731   end
1732 end
1733 </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 594, 627, 651, 685, 707, 737, 776

bool commands:

\bool_if:NTF

| | | | |
|-------------------------|---|-----------------------------|--|
| \bool_lazy_and:nnTF | 1430 | \dim_to_decimal_in_sp:n | 1211, 1216, 1221 |
| \bool_new:N | 481 | \c_zero_dim | 1104, 1105, 1106, 1268, 1269, 1270 |
| \bool_set_true:N | 917, 999, 1092, 1194 | \directlua | 59, 1501 |
| box commands: | | E | |
| \box_dp:N | 930, 1011, 1103, 1207, 1221 | exp commands: | |
| \box_ht:N | 927, 1008, 1100, 1204, 1216 | \exp_args:Ne | 1350, 1356, 1401, 1407, 1421, 1451, 1456, 1481, 1486, 1520, 1525, 1550, 1555 |
| \box_new:N | 50, 51, 1089 | \exp_args:NNx | 797 |
| \box_scale:Nnn | 1225 | \exp_args:Nnx | 450, 629, 653 |
| \box_set_dp:Nn | 1104, 1171, 1268, 1292 | \exp_args:Nxxx | 709, 724, 739, 754 |
| \box_set_ht:Nn | 1105, 1170, 1269, 1291 | \exp_args:Nx | 202, 313, 352, 625, 635, 649, 659, 703, 733, 1148 |
| \box_set_wd:Nn | 1106, 1169, 1270, 1290 | \exp_not:n | 1333, 1354, 1405 |
| \box_use:N | 1274 | F | |
| \box_use_drop:N | 1117, 1172, 1249, 1293 | fp commands: | |
| \box_wd:N | 924, 1005, 1097, 1201, 1211 | \fp_eval:n | 1347, 1368, 1449, 1469, 1518, 1538 |
| C | | H | |
| clist commands: | | hbox commands: | |
| \clist_const:Nn | 379 | \hbox:n | 1378, 1389 |
| \clist_map_function:NN | 810 | \hbox_gset:Nn | 1090 |
| \clist_map_inline:Nn | 388, 422, 438, 607 | \hbox_set:Nn | 915, 997, 1162, 1192, 1226, 1281 |
| cs commands: | | hook commands: | |
| \cs_generate_variant:Nn | 29, 30, 41, 42 | \hook_gput_code:nnn | 104, 441, 1271 |
| \cs_gset_eq:NN | 595, 596, 686, 687, 777, 778, 1318, 1319, 1320 | \hook_gput_next_code:nn | 1107 |
| \cs_if_exist:NTF | 391, 873 | \hook_gset_rule:nnnn | 435, 436 |
| \cs_new:Npn | 37, 62, 68, 210, 786, 983, 1065, 1155, 1179, 1295 | I | |
| \cs_new_protected:Npn | 31, 112, 121, 137, 143, 149, 156, 163, 172, 192, 215, 225, 239, 251, 268, 279, 286, 293, 302, 311, 318, 325, 332, 341, 350, 358, 361, 367, 372, 375, 403, 414, 420, 443, 447, 460, 463, 464, 468, 471, 472, 476, 510, 526, 605, 695, 795, 819, 826, 833, 842, 846, 849, 852, 864, 869, 870, 909, 976, 991, 1056, 1079, 1160, 1177, 1178, 1185, 1279, 1316, 1373, 1570 | int commands: | |
| \cs_new_protected:Npx | 131 | \int_compare:nNnTF | 886, 938, 1019, 1501 |
| \cs_set_eq:NN | 1324, 1325, 1326 | \int_compare_p:nNn | 1431, 1432 |
| \cs_set_protected:Npn | 490, 494, 498, 502, 506, 516, 518, 520, 522, 524, 537, 556, 575, 581, 587, 592, 599, 622, 646, 670, 674, 679, 683, 690, 700, 730, 761, 765, 770, 774, 781, 875, 879, 888, 892, 899, 903, 1329, 1419, 1424, 1434, 1473, 1494, 1503, 1542, 1563 | \int_const:Nn | 971, 1051, 1086, 1188 |
| D | | \int_gincr:N | 175, 539, 558, 624, 648, 702, 706, 732, 736, 1085, 1187 |
| dim commands: | | \int_if_exist:NTF | 1305 |
| \dim_eval:n | 1422, 1478, 1479, 1480, 1489, 1490, 1491, 1547, 1548, 1549, 1558, 1559, 1560 | \int_new:N | 54, 55, 56 |
| | | \int_use:N | 176, 179, 542, 550, 561, 569, 626, 631, 640, 650, 655, 664, 704, 711, 715, 718, 726, 734, 741, 745, 748, 756, 979, 985, 1058, 1066, 1157, 1235, 1262, 1286, 1297, 1455, 1485, 1524, 1554 |
| | | K | |
| | | kernel internal commands: | |
| | | __kernel_backend_literal:n | 45, 540, 544, 559, 563, 577, 589, 601, 611, 1576, 1577, 1581, 1589 |

| | |
|---|---|
| <pre> __kernel_backend_literal_page:n 625, 649, 672, 681, 692, 703, 733, 763, 772, 783 __kernel_backend_postscript:n 1228, 1250, 1256, 1283 __kernel_kern:n 1377, 1385, 1388, 1417 __kernel_pdf_name_from_unicode_- e:n 62, 68 __kernel_pdffdict_name:n 194, 195, 197, 425, 451, 609, 789, 800, 805, 918, 939, 950, 955, 960, 965, 1000, 1020, 1030, 1035, 1040, 1045, 1195 \g__kernel_pdfmanagement_end_- run_code_tl 77, 84, 91 \g__kernel_pdfmanagement_- thispage_shipout_code_tl 100, 106 </pre> | <pre> __pdf_backend_bdc_contstream:nn 587, 596, 679, 687, 770, 778 __pdf_backend_bdcobject:n 12, 478, 498, 520, 556, 584, 646, 677, 730, 768 __pdf_backend_bdcobject:nn 12, 478, 494, 518, 537, 622, 700 __pdf_backend_bmc:n 12, 478, 506, 524, 575, 670, 761 __pdf_backend_catalog_gput:nn .. 18 __pdf_backend_destination:nn 1318, 1324 __pdf_backend_destination:nnnn 1319, 1325, 1473, 1542 __pdf_backend_emc: 12, 478, 502, 522, 599, 690, 781 __pdf_backend_link_begin:n .. 1426 __pdf_backend_link_begin:nnnw 1496, 1565 __pdf_backend_link_begin_- goto:nnw 1320, 1326 __pdf_backend_link_begin_- structure_goto:nnw 1320, 1326, 1424, 1494, 1563 __pdf_backend_link_off: 869, 875, 888, 899 __pdf_backend_link_on: 870, 879, 892, 903 __pdf_backend_luastring:n 125, 210, 219, 231, 232, 243, 258, 259 \g__pdf_backend_name_int 53, 539, 542, 550, 558, 561, 569, 624, 626, 631, 640, 648, 650, 655, 664, 702, 704, 732, 734 __pdf_backend_Names_gpush:nn 819, 826, 833, 842, 846 __pdf_backend_NamesEmbeddedFiles_- add:nn 848, 849, 852, 864 \g__pdf_backend_object_int 1085, 1088, 1187, 1190, 1235 __pdf_backend_object_last: 500, 570, 656, 665, 742, 757 __pdf_backend_object_ref:n 397, 456, 496, 551, 614, 632, 641, 712, 727 __pdf_backend_Page_gput:nn 5, 146, 156, 225, 286, 325, 361 __pdf_backend_Page_gremove:n 5, 146, 163, 239, 293, 332, 367 \g__pdf_backend_page_int 53 __pdf_backend_Page_primitive:n 5, 146, 149, 202, 215, 279, 304, 313, 318, 343, 352, 358 __pdf_backend_PageResources:n 443, 463, 471 </pre> |
| L | |
| <pre> lualua commands: \lualua: 169, 248, 299, 338 </pre> | |
| M | |
| <pre> mode commands: \mode_leave_vertical: ... 1109, 1273 </pre> | |
| P | |
| <pre> pdf commands: \pdf_activate_structure_destination: 1315, 1316 \l_pdf_current_structure_- destination_tl 1312, 1350, 1356, 1401, 1407, 1451, 1456, 1481, 1486, 1520, 1525, 1550, 1555 \pdf_object_if_exist:nTF 1350, 1401, 1451, 1481, 1520, 1550 \pdf_object_new:nn 390, 440 \pdf_object_ref:n .. 791, 952, 957, 962, 967, 1032, 1037, 1042, 1047, 1123, 1130, 1137, 1145, 1356, 1407 \pdf_object_ref_last: . 822, 829, 836 \pdf_object_unnamed_write:nn 583, 676, 767, 821, 828, 835 \pdf_object_write 453 \pdf_object_write:nn 427, 441 </pre> | |
| <pre> pdf internal commands: __pdf_backend:n 139, 445, 454, 836, 901, 905, 1110, 1118, 1119, 1126, 1133, 1140, 1149, 1164, 1331, 1352, 1380, 1381, 1391, 1403 __pdf_backend_bdc:nn 12, 478, 490, 516, 592, 595, 596, 597, 683, 686, 687, 688, 774, 777, 778, 779 __pdf_backend_bdc_contobj:nn 581, 595, 674, 686, 765, 777 </pre> | |

```

\c__pdf_backend_PageResources_-
  clist .. 378, 388, 422, 438, 607, 811
\__pdf_backend_PageResources_-
  gpush:n .....
  .... 12, 478, 510, 526, 605, 695, 795
\__pdf_backend_PageResources_-
  gpush_aux:n ..... 786, 812
\__pdf_backend_PageResources_-
  gput:nnn 387, 403, 414, 447, 464, 472
\__pdf_backend_PageResources_-
  obj_gpush: . 387, 420, 460, 468, 476
\__pdf_backend_Pages_primitive:n
  ..... 111, 112, 121, 131, 137, 143
\__pdf_backend_pdfmark:n .....
  ..... 492, 496, 500, 504, 508, 854
\__pdf_backend_ref_label:nn ....
  ..... 31, 41, 176, 715, 745
\__pdf_backend_ref_value:nn ....
  ..... 37, 42, 179, 718, 748
\g__pdf_backend_resourceid_int ..
  .... 53, 175, 176, 179, 706, 711,
  715, 718, 726, 736, 741, 745, 748, 756
\__pdf_backend_set_regression_-
  data: ..... 1570
\__pdf_backend_structure_-
  destination:nn .....
  .. 1318, 1324, 1328, 1329, 1434, 1503
\__pdf_backend_structure_-
  destination:nnnn 1319, 1325, 1419
\__pdf_backend_structure_-
  destination_aux:nnnn .. 1373, 1421
\__pdf_backend_ThisPage_gpush:n .
  .... 5, 146, 192, 268, 311, 350, 375
\__pdf_backend_ThisPage_gput:nn .
  .... 5, 146, 172, 251, 302, 341, 372
\g__pdf_backend_thispage_-
  shipout_tl ..... 5
\l__pdf_backend_tmpa_box .....
  .... 47, 915, 924, 927, 930, 970,
  997, 1005, 1008, 1011, 1050, 1162,
  1169, 1170, 1171, 1172, 1192, 1201,
  1204, 1207, 1211, 1216, 1221, 1225,
  1249, 1281, 1290, 1291, 1292, 1293
\l__pdf_backend_tmpb_box .....
  .... 51, 1226, 1268, 1269, 1270, 1274
\l__pdf_backend_xform_bool . 481,
  627, 651, 707, 737, 917, 999, 1092, 1194
\__pdf_backend_xform_if_exist:n .
  ..... 1303, 1309
\__pdf_backend_xform_new:nnnn ...
  .... 908, 909, 991, 1079, 1177, 1185
\__pdf_backend_xform_ref:n . 908,
  983, 1065, 1112, 1155, 1166, 1179, 1295

\l__pdf_backend_xform_tmpdp_tl ..
  ..... 1183, 1219, 1233, 1240
\l__pdf_backend_xform_tmpht_tl ..
  ..... 1184, 1214, 1238
\l__pdf_backend_xform_tmpwd_tl ..
  ..... 1182, 1209, 1239
\__pdf_backend_xform_use:n .....
  .... 908, 976, 1056, 1160, 1178, 1279
\g__pdf_tmpa_prop ... 47, 194, 199, 204
\l__pdf_tmpa_tl .....
  .... 47, 177, 181, 183, 186, 716,
  720, 722, 725, 746, 750, 752, 755, 758

pdfdict commands:
\pdfdict_gput:nnn .....
  . 158, 186, 288, 327, 363, 405, 416,
  466, 474, 629, 653, 709, 724, 739, 754
\pdfdict_gremove:nn 165, 295, 334, 369
\pdfdict_if_exist:nTF . 181, 720, 750
\pdfdict_item:nn ..... 204, 791, 806
\pdfdict_new:n ..... 183, 722, 752
\pdfdict_show:n ..... 758
\pdfdict_use:n 314, 353, 429, 945, 1026
\pdfextension ..... 890, 894

pdfmanagement commands:
\pdfmanagement_add:nnn .... 1573,
  1580, 1588, 1596, 1601, 1610, 1611

pdfmanagement internal commands:
\g__pdfmanagement_active_bool ...
  ..... 594, 685, 776
\pdfnames ..... 18
\pdfpageref ..... 2
\pdfrunninglinkoff ..... 873, 877
\pdfrunninglinkon ..... 881
\pdftrailerid ..... 1598

pdfxform commands:
\pdfxform_dp:n ..... 1115, 1171, 1292
\pdfxform_ht:n ..... 1114, 1170, 1291
\pdfxform_if_exist:n ..... 1309
\pdfxform_wd:n ..... 1113, 1169, 1290

prg commands:
\prg_new_conditional:Npnn .... 1303
\prg_new_eq_conditional:NNn .. 1309
\prg_return_false: ..... 1307
\prg_return_true: ..... 1306

prop commands:
\prop_count:N ..... 939, 1020
\prop_gclear:N ..... 918, 1000, 1195
\prop_gput:Nnn ..... 199, 451
\prop_gset_eq:NN ..... 194
\prop_if_empty:NTF .....
  ..... 424, 609, 788, 949,
  954, 959, 964, 1029, 1034, 1039, 1044
\prop_if_exist:NTF ..... 195, 799
\prop_map_function:NN ..... 204, 804

```


| | | | |
|---|---|----------------------------|--|
| \prop_map_inline:Nn | 197 | \tex luatexversion:D | 886 |
| \prop_new:N | 48 | \tex_pdfdest:D | 1436, 1453, 1475, 1483 |
| \ProvidesExplFile | 1 | \tex_pdfextension:D | 829, 1505, 1522, 1544, 1552 |
| R | | \tex_pdflastxform:D | 973, 1053 |
| ref commands: | | \tex_pdfnames:D | 822 |
| \ref_label:nn | 29, 34 | \tex_pdfpageattr:D | 151 |
| \ref_value:nn | 30, 39 | \tex_pdfpageresources:D | 797 |
| \relax | 97, 1602 | \tex_pdfpagesattr:D | 114 |
| \RequirePackage | 28 | \tex_pdfrefxform:D | 978, 1058 |
| S | | \tex_pdfsuppressptexinfo:D | 1597 |
| scan commands: | | \tex_pdftexrevision:D | 1432 |
| \scan_stop: | 980, 1062, 1450, 1470, 1480, 1491, 1519, 1539, 1549, 1560, 1597 | \tex_pdftexversion:D | 1431 |
| str commands: | | \tex_pdfvariable:D | 1602, 1603 |
| \str_case:nnTF | 1336, 1357, 1438, 1458, 1507, 1527 | \tex_pdfxform:D | 932, 1013 |
| \str_convert_pdfname:n | 64, 452 | \tex_special:D | 133, 281, 320 |
| \str_if_eq:nnTF | 1243, 1254 | \tex_the:D | 924, 927, 930, 1005, 1008, 1011, 1097, 1100, 1103, 1201, 1204, 1207 |
| sys commands: | | \tex_unexpanded:D | 212 |
| \sys_gset_rand_seed:n | 1572 | \tex_vss:D | 1383, 1415 |
| \sys_if_engine luatex:TF | 119 | text commands: | |
| T | | \text_expand:n | 64, 70 |
| T _E X and L ^A T _E X 2 _ε commands: | | tl commands: | |
| \@bsphack | 33 | \c_space_tl | 542, 550, 561, 569, 626, 650, 704, 734, 1113, 1114, 1115, 1235 |
| \@esphack | 35 | \tl_const:Nn | 922, 925, 928, 1003, 1006, 1009, 1095, 1098, 1101, 1199, 1202, 1205 |
| \@kernel@after@enddocument@afterlastpage | 74, 75 | \tl_gput_right:Nn | 75, 82, 89 |
| \@kernel@after@shipout@background | 95, 98 | \tl_if_exist:NTF | 95 |
| \@kernel@after@shipout@lastpage | 81, 82, 88, 89 | \tl_new:N | 49, 1182, 1183, 1184, 1313 |
| \@kernel@before@shipout@background | 97 | \tl_set:Nn | 177, 716, 746, 1209, 1214, 1219 |
| \g@addto@macro | 97, 98 | \tl_to_str:n | 923, 926, 929, 972, 979, 985, 1004, 1007, 1010, 1052, 1060, 1066, 1087, 1096, 1099, 1102, 1157, 1189, 1200, 1203, 1206, 1262, 1286, 1297, 1305, 1456, 1486, 1525, 1555 |
| tex commands: | | \tl_use:N | 1233, 1238, 1239, 1240 |
| \tex_directlua:D | 123, 227, 241, 391, 393 | V | |
| \tex_global:D | 114, 151, 797 | vbox commands: | |
| \tex_immediate:D | 932, 1013 | \vbox to zero:n | 1375, 1386 |
| \tex_latelua:D | 217, 253, 270, 406, 407, 635, 659 | | |
| \tex luaescapestring:D | 212 | | |