The l3backend-testphase package
Additional backend PDF features
LaTeX PDF management testphase bundle

The LaTeX Project*

Version 0.96l, released 2024-09-13

1 l3backend-testphase Implementation

```
1 ⟨drivers⟩\ProvidesExplFile
2 ⟨∗dvipdfmx⟩
3   {l3backend-testphase-dvipdfmx.def}{2024-09-13}{}
4   {LaTeX~PDF~management~testphase~bundle~backend~support: dvipdfmx}
5 ⟨/dvipdfmx⟩
6 ⟨∗dvips⟩
7   {l3backend-testphase-dvips.def}{2024-09-13}{}
8   {LaTeX~PDF~management~testphase~bundle~backend~support: dvips}
9 ⟨/dvips⟩
10 ⟨∗dvisvgm⟩
11   {l3backend-testphase-dvisvgm.def}{2024-09-13}{}
12   {LaTeX~PDF~management~testphase~bundle~backend~support: dvisvgm}
13 ⟨/dvisvgm⟩
14 ⟨∗luatex⟩
15   {l3backend-testphase-luatex.def}{2024-09-13}{}
16   {LaTeX~PDF~management~testphase~bundle~backend~support: PDF output (LuaTeX)}
17 ⟨/luatex⟩
18 ⟨∗pdftex⟩
19   {l3backend-testphase-pdftex.def}{2024-09-13}{}
20   {LaTeX~PDF~management~testphase~bundle~backend~support: PDF output (pdfTeX)}
21 ⟨/pdftex⟩
22 ⟨∗xdvipdfmx⟩
23   {l3backend-testphase-xetex.def}{2024-09-13}{}
24   {LaTeX~PDF~management~testphase~bundle~backend~support: XeTeX}
25 ⟨/xdvipdfmx⟩
```

1.1 Variants

We need to generate temporarily a few e-types variants of kernel backend commands. These can be removed once the kernel provides them.

```
26 ⟨@@=pdf⟩
27 ⟨∗luatex | pdftex⟩
28 \cs_generate_variant:Nn \__kernel_backend_literal_page:n { e }
```

---

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

```
29 ⟨/luatex | pdftex⟩
30 ⟨*dvipdfmx | xdvipdfmx⟩
31 \cs_generate_variant:Nn \__kernel_backend_literal:n { e }
32 \cs_generate_variant:Nn \__pdf_backend:n { e }
33 ⟨/dvipdfmx | xdvipdfmx⟩
34 ⟨*dvips⟩
35 \cs_generate_variant:Nn \__kernel_backend_postscript:n { e }
36 \cs_generate_variant:Nn \__pdf_backend_pdfmark:n { e }
37 ⟨/dvips⟩
```

1.2 Support for delayed literal and special

Starting with TeXlive 2023 the engines support a `shipout` keyword for `\pdfliteral` and `\special`. When used the argument is not expanded when the command is used but only when the page is shipped out. This allows for example the tagging code to delay the page-wise numbering of MC-chunks until the page is actually built. For now we test the engine support. The boolean is setup in pdfmanagement-testphase.dtx.

```
38 ⟨*drivers⟩
```

The following commands provide the needed kernel backend support. This are basically copies of similar commands of l3backend-basics.

__kernel_backend_shipout_literal:e   The one shared function for all backends is access to the basic `\special` primitive.

```
39 \bool_if:NT \l__pdfmanagement_delayed_shipout_bool
40   {
41     \cs_new_protected:Npn \__kernel_backend_shipout_literal:e #1
42       { \tex_special:D~shipout { #1} }
43 ⟨/drivers⟩
```

(*End of definition for* __kernel_backend_shipout_literal:e.)

```
44 ⟨*luatex | pdftex⟩
```

__kernel_backend_shipout_literal_pdf:e   This is equivalent to `\special{pdf:}` but the engine can track it. Without the `direct` keyword everything is kept in sync: the transformation matrix is set to the current point automatically. Note that this is still inside the text (BT ... ET block).

```
45   \cs_new_protected:Npn \__kernel_backend_shipout_literal_pdf:e #1
46     {
47 ⟨*luatex⟩
48       \tex_pdfextension:D ~ literal ~ shipout ~
49 ⟨/luatex⟩
50 ⟨*pdftex⟩
51       \tex_pdfliteral:D ~ shipout ~
52 ⟨/pdftex⟩
53         { #1 }
54     }
```

(*End of definition for* __kernel_backend_shipout_literal_pdf:e.)

__kernel_backend_shipout_literal_page:e   Page literals are pretty simple.

```
55   \cs_new_protected:Npn \__kernel_backend_shipout_literal_page:e #1
56     {
57 ⟨*luatex⟩
58       \tex_pdfextension:D ~ literal ~ shipout ~
59 ⟨/luatex⟩
```

```
60 ⟨∗pdftex⟩
61         \tex_pdfliteral:D ~ shipout ~
62 ⟨/pdftex⟩
63             page { #1 }
64     }
65 ⟨/luatex | pdftex⟩
66 ⟨drivers⟩ }
```

(*End of definition for* __kernel_backend_shipout_literal_page:e.)

1.3   Crossreferences

Commands to get a reference for the absolute page counter.

```
67 ⟨∗drivers⟩
68 \cs_new_protected:Npn \__pdf_backend_record_abspage:n #1
69   {
70       \@bsphack
71       \property_record:nn{#1}{abspage}
72       \@esphack
73   }
74 \cs_new:Npn \__pdf_backend_ref_abspage:n #1
75   {
76       \property_ref:nn{#1}{abspage}
77   }
78
79 \cs_generate_variant:Nn \__pdf_backend_record_abspage:n {e}
80 \cs_generate_variant:Nn \__pdf_backend_ref_abspage:n {e}
81 ⟨/drivers⟩
```

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

```
82 ⟨∗dvipdfmx | xdvipdfmx⟩
83       \__kernel_backend_literal:n { dvipdfmx:config~C~ 0x0010 }
84 ⟨/dvipdfmx | xdvipdfmx⟩
```

\g__pdf_tmpa_prop
\l__pdf_tmpa_tl
\l__pdf_backend_tmpa_box

Some scratch variables

```
85 ⟨∗drivers⟩
86 \prop_new:N \g__pdf_tmpa_prop
87 \tl_new:N   \l__pdf_tmpa_tl
88 \box_new:N  \l__pdf_backend_tmpa_box
89 \box_new:N  \l__pdf_backend_tmpb_box
90 ⟨/drivers⟩
```

(*End of 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.

```
91 ⟨∗drivers⟩
92 \int_new:N \g__pdf_backend_resourceid_int
93 \int_new:N \g__pdf_backend_name_int
94 \int_new:N \g__pdf_backend_page_int
95 ⟨/drivers⟩
```

(*End of definition for* \g__pdf_backend_resourceid_int, \g__pdf_backend_name_int, *and* \g__pdf_-
backend_page_int.)

1.4   luacode

Load the lua code.

```
96  ⟨*luatex⟩
97      \directlua { require("l3backend-testphase.lua") }
98  ⟨/luatex⟩
```

1.5   Converting unicode strings to a pdfname

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

```
99   ⟨*pdftex | luatex | dvipdfmx | xdvipdfmx | dvisvgm⟩
100  \cs_new:Npn \__kernel_pdf_name_from_unicode_e:n #1
101    {
102      / \str_convert_pdfname:e { \text_expand:n { #1 } } }
103    }
104  ⟨/pdftex | luatex | dvipdfmx | xdvipdfmx | dvisvgm⟩
105  ⟨*dvips⟩
106  \cs_new:Npn \__kernel_pdf_name_from_unicode_e:n #1
107    {
108      ~ ( \text_expand:n { #1 } ) ~ cvn
109    }
110  ⟨/dvips⟩
```

1.6   Hooks

1.6.1   Add the "end run" hooks

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

```
111  ⟨*pdftex | luatex⟩
112  % put in \@kernel@after@enddocument@afterlastpage
113  \tl_gput_right:Nn \@kernel@after@enddocument@afterlastpage
114    {
115      \g__kernel_pdfmanagement_end_run_code_tl
116    }
117  ⟨/pdftex | luatex⟩
118  ⟨*dvipdfmx | xdvipdfmx⟩
119  % put in \@kernel@after@shipout@lastpage
120  \tl_gput_right:Nn \@kernel@after@shipout@lastpage
121    {
122      \g__kernel_pdfmanagement_end_run_code_tl
123    }
124  ⟨/dvipdfmx | xdvipdfmx⟩
125  ⟨*dvips⟩
126  % put in \@kernel@after@shipout@lastpage
127  \tl_gput_right:Nn\@kernel@after@shipout@lastpage
128    {
129      \g__kernel_pdfmanagement_end_run_code_tl
130    }
131  ⟨/dvips⟩
```

1.6.2 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 ...

```
132  ⟨*drivers⟩
133  \tl_if_exist:NTF \@kernel@after@shipout@background
134    {
135      \g@addto@macro \@kernel@before@shipout@background{\relax}
136      \g@addto@macro \@kernel@after@shipout@background
137        {
138          \g__kernel_pdfmanagement_thispage_shipout_code_tl
139        }
140    }
141    {
142      \hook_gput_code:nnn{shipout/background}{pdf}
143        {
144          \g__kernel_pdfmanagement_thispage_shipout_code_tl
145        }
146    }
147
148  ⟨/drivers⟩
```

1.7   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.

```
149  ⟨*pdftex⟩
150  \cs_new_protected:Npn \__pdf_backend_Pages_primitive:n #1
151    {
152      \tex_global:D \tex_pdfpagesattr:D { #1 }
153    }
154  ⟨/pdftex⟩
155  ⟨*luatex⟩
156  %luatex: does it in lua
157  \sys_if_engine_luatex:T
158    {
159      \cs_new_protected:Npn \__pdf_backend_Pages_primitive:n #1
160        {
161          \tex_directlua:D
162            {
163              pdf.setpagesattributes( \__pdf_backend_luastring:n { #1 } )
164            }
165        }
166    }
167  ⟨/luatex⟩
168  ⟨*dvips⟩
169  \cs_new_protected:Npx \__pdf_backend_Pages_primitive:n #1
170    {
171      \tex_special:D{ps:~[#1~/PAGES~pdfmark} %]
172    }
```

```
173 ⟨/dvips⟩
174 ⟨∗dvipdfmx | xdvipdfmx⟩
175 \cs_new_protected:Npn \__pdf_backend_Pages_primitive:n #1
176   {
177     \__pdf_backend:n{put~@pages~<<#1>>}
178   }
179 ⟨/dvipdfmx | xdvipdfmx⟩
180 ⟨∗dvisvgm⟩
181 \cs_new_protected:Npn \__pdf_backend_Pages_primitive:n #1
182   {}
183 ⟨/dvisvgm⟩
```

(*End of definition for* `\__pdf_backend_Pages_primitive:n`.)

1.8 "Page" and "ThisPage" attributes (pdfpageattr)

<div style="float:left">

`\__pdf_backend_Page_primitive:n`
`\__pdf_backend_Page_gput:nn`
`\__pdf_backend_Page_gremove:n`
`\__pdf_backend_ThisPage_gput:nn`
`\__pdf_backend_ThisPage_gpush:n`

</div>

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

```
184 %  backend commands
185 ⟨∗pdftex⟩
186  %the primitive
187   \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
188     {
189       \tex_global:D \tex_pdfpageattr:D { #1 }
190     }
191 % the command to store default values.
192 % Uses a prop with pdflatex + dvi,
193 % sets a lua table with lualatex
194  \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2 %key,value
195    {
196      \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
197    }
198 % the command to remove a default value.
199 % Uses a prop with pdflatex + dvi,
200 % changes a lua table with lualatex
201 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
202   {
203     \pdfdict_gremove:nn  {g__pdf_Core/Page}{ #1 }
204   }
205 % the command used in the document.
206 % direct call of the primitive special with dvips/dvipdfmx
207 % \latelua: fill a page related table with lualatex, merge it with the page
208 % table and push it directly
209 % write to aux and store in prop with pdflatex
210 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
211   {
212     %we need to know the page the resource should be added too.
```

```
213    \int_gincr:N\g__pdf_backend_resourceid_int
214    \__pdf_backend_record_abspage:e { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
215    \tl_set:Ne \l__pdf_tmpa_tl
216      {
217        \__pdf_backend_ref_abspage:e {l3pdf\int_use:N\g__pdf_backend_resourceid_int}
218      }
219    \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl}
220      {
221        \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl}
222      }
223    %backend_Page has no handler.
224    \pdfdict_gput:nnn {g__pdf_Core/backend_Page\l__pdf_tmpa_tl}{ #1 }{ #2 }
225  }
226  %the code to push the values, used in shipout
227  %merges the two props and then fills the register in pdflatex
228  %merges the two tables and then fills (in lua) in luatex
229  %issues the values stored in the global prop with dvi
230  \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
231    {
232      \prop_gset_eq:Nc \g__pdf_tmpa_prop { \__kernel_pdfdict_name:n { g__pdf_Core/Page } }
233      \prop_if_exist:cT  { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1 } }
234        {
235          \prop_map_inline:cn { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1 } }
236            {
237              \prop_gput:Nnn \g__pdf_tmpa_prop { ##1 }{ ##2 }
238            }
239        }
240      \__pdf_backend_Page_primitive:e
241        {
242          \prop_map_function:NN \g__pdf_tmpa_prop \pdfdict_item:ne
243        }
244    }
245  ⟨/pdftex⟩
246  ⟨*luatex⟩
247  % do we need to use some escaping for the values?????
248  \cs_new:Npn \__pdf_backend_luastring:n #1
249    {
250      "\tex_luaescapestring:D { \tex_unexpanded:D { #1 } }"
251    }
252   %not used, only there for consistency
253  \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
254    {
255      \tex_latelua:D
256        {
257          pdf.setpageattributes(\__pdf_backend_luastring:n { #1 })
258        }
259    }
260    % the command to store default values.
261    % Uses a prop with pdflatex + dvi,
262    % sets a lua table with lualatex
263  \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
264    {
265      \tex_directlua:D
266        {
```

```
267        ltx.__pdf.backend_Page_gput
268          (
269            \__pdf_backend_luastring:n { #1 },
270            \__pdf_backend_luastring:n { #2 }
271          )
272      }
273  }
274  % the command to remove a default value.
275  % Uses a prop with pdflatex + dvi,
276  % changes a lua table with lualatex
277 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
278  {
279    \tex_directlua:D
280      {
281        ltx.__pdf.backend_Page_gremove (\__pdf_backend_luastring:n { #1 })
282      }
283  }
284  % the command used in the document.
285  % direct call of the primitive special with dvips/dvipdfmx
286  % \latelua: fill a page related table with lualatex, merge it with the page
287  % table and push it directly
288  % write to aux and store in prop with pdflatex
289 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
290  {
291    \tex_latelua:D
292      {
293        ltx.__pdf.backend_ThisPage_gput
294          (
295            tex.count["g_shipout_readonly_int"],
296            \__pdf_backend_luastring:n { #1 },
297            \__pdf_backend_luastring:n { #2 }
298          )
299        ltx.__pdf.backend_ThisPage_gpush (tex.count["g_shipout_readonly_int"])
300      }
301  }
302  %the code to push the values, used in shipout
303  %merges the two props and then fills the register in pdflatex
304  %merges the two tables (the one is probably still empty) and then fills (in lua) in luatex
305  %issues the values stored in the global prop with dvi
306 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
307  {
308    \tex_latelua:D
309      {
310        ltx.__pdf.backend_ThisPage_gpush (tex.count["g_shipout_readonly_int"])
311      }
312  }
313
314 ⟨/luatex⟩
315 ⟨*dvipdfmx | xdvipdfmx⟩
316  %the primitive
317 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
318  {
319    \tex_special:D{pdf:~put~@thispage~<<#1>>}
320  }
```

```
321  % the command to store default values.
322  % Uses a prop with pdflatex + dvi,
323  % sets a lua table with lualatex
324 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
325    {
326      \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
327    }
328  % the command to remove a default value.
329  % Uses a prop with pdflatex + dvi,
330  % changes a lua table with lualatex
331 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
332    {
333      \pdfdict_gremove:nn  {g__pdf_Core/Page}{ #1 }
334    }
335  % the command used in the document.
336  % direct call of the primitive special with dvips/dvipdfmx
337  % \latelua: fill a page related table with lualatex, merge it with the page
338  % table and push it directly
339  % write to aux and store in prop with pdflatex
340 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
341    {
342      \__pdf_backend_Page_primitive:n { /#1~#2 }
343    }
344  %the code to push the values, used in shipout
345  %merges the two props and then fills the register in pdflatex
346  %merges the two tables (the one is probably still empty)
347  % and then fills (in lua) in luatex
348  %issues the values stored in the global prop with dvi
349 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
350    {
351      \__pdf_backend_Page_primitive:e
352        { \pdfdict_use:n { g__pdf_Core/Page} }
353    }
354 ⟨/dvipdfmx | xdvipdfmx⟩
355 ⟨∗dvips⟩
356 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
357     {
358      \tex_special:D{ps:~[{ThisPage}<<#1>>~/PUT~pdfmark} %]
359     }
360  % the command to store default values.
361  % Uses a prop with pdflatex + dvi,
362  % sets a lua table with lualatex
363 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
364     {
365      \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
366     }
367  % the command to remove a default value.
368  % Uses a prop with pdflatex + dvi,
369  % changes a lua table with lualatex
370 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
371     {
372      \pdfdict_gremove:nn  {g__pdf_Core/Page}{ #1 }
373     }
374  % the command used in the document.
```

```
375    % direct call of the primitive special with dvips/dvipdfmx
376    % \latelua: fill a page related table with lualatex, merge it with the page
377    % table and push it directly
378    % write to aux and store in prop with pdflatex
379  \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
380    {
381      \__pdf_backend_Page_primitive:n { /#1~#2 }
382    }
383    %the code to push the values, used in shipout
384    %merges the two props and then fills the register in pdflatex
385    %merges the two tables (the one is probably still empty)
386    %and then fills (in lua) in luatex
387    %issues the values stored in the global prop with dvi
388  \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
389    {
390      \__pdf_backend_Page_primitive:e
391        { \pdfdict_use:n { g__pdf_Core/Page} }
392    }
393  ⟨/dvips⟩
394  ⟨∗dvisvgm⟩
395  % mostly only dummies ...
396  \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
397      {}
398    % Uses a prop with pdflatex + dvi,
399  \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
400    {
401      \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
402    }
403    % the command to remove a default value.
404    % Uses a prop with pdflatex + dvi,
405  \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
406    {
407      \pdfdict_gremove:nn  {g__pdf_Core/Page}{ #1 }
408    }
409    % the command used in the document.
410  \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
411      {}
412    %the code to push the values, used in shipout
413  \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
414      {}
415  ⟨/dvisvgm⟩
416  ⟨∗drivers⟩
417  \cs_generate_variant:Nn \__pdf_backend_Page_primitive:n { e }
418  ⟨/drivers⟩
```

(*End of definition for* __pdf_backend_Page_primitive:n *and others.*)

1.9 "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.

```
419 ⟨*drivers⟩
420 \clist_const:Nn \c__pdf_backend_PageResources_clist
421   {
422     ExtGState,
423     ColorSpace,
424     Pattern,
425     Shading,
426   }
427 ⟨/drivers⟩
```

(*End of 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

__pdf_backend_PageResources_obj_gpush: issued in the end-of-run hook! create the backend objects:

```
428 ⟨*pdftex | luatex⟩
429 \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
430   {
431     \pdf_object_new:n {__pdf/Page/Resources/#1}
432     \cs_if_exist:NT \tex_directlua:D
433       {
434         \tex_directlua:D
435           {
436             ltx.__pdf.object["__pdf/Page/Resources/#1"]
437             =
438             "\pdf_object_ref:n{__pdf/Page/Resources/#1}"
439           }
440       }
441   }
442 ⟨/pdftex | luatex⟩
```

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

```
443 ⟨*luatex⟩
444 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
445   {
446     \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
447     \tex_latelua:D{ltx.__pdf.Page.Resources.#1=true}
448     \tex_latelua:D
449       {
450         ltx.pdf.Page_Resources_gpush(tex.count["g_shipout_readonly_int"])
451       }
452   }
453 ⟨/luatex⟩
454 ⟨*pdftex⟩
455 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
456   {
```

```
457        \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
458      }
459  ⟨/pdftex⟩
```

code for end of document code

```
460  ⟨*pdftex | luatex⟩
461  \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush:
462    {
463      \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
464        {
465          \prop_if_empty:cF
466            { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/##1} }
467            {
468              \pdf_object_write:nne
469                { __pdf/Page/Resources/##1 } { dict }
470                { \pdfdict_use:n { g__pdf_Core/Page/Resources/##1} }
471            }
472        }
473    }
474  ⟨/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.

```
475  ⟨*dvipdfmx | xdvipdfmx⟩
476  ⟨xdvipdfmx⟩\hook_gset_rule:nnnn{shipout/firstpage}{l3backend-xetex}{after}{pdf}
477  ⟨dvipdfmx⟩\hook_gset_rule:nnnn{shipout/firstpage}{l3backend-dvipdfmx}{after}{pdf}
478  %
479  \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
480    {
481      \pdf_object_new:n    { __pdf/Page/Resources/#1 }
482      \hook_gput_code:nnn
483        {shipout/firstpage}
484        {pdf}
485        {\pdf_object_write:nnn { __pdf/Page/Resources/#1 } { dict } {}}
486    }
487  \cs_new_protected:Npn \__pdf_backend_PageResources:n #1
488    {
489      \__pdf_backend:n {put~@resources~<<#1>>}
490    }
491  \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
492    {
493      % this is not used for output, but there is a test if the resource is empty
494      \prop_gput:cne { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/#1} }
495        { \str_convert_pdfname:n {#2} }{ #3 }
496      %objects are not filled with \pdf_object_write as this is not additive!
497      \__pdf_backend:e
498        {
499          put~\pdf_object_ref:n {__pdf/Page/Resources/#1}<</#2~#3>>
500        }
501    }
502
503  \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
```

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.

```
505 ⟨∗dvips⟩
506 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1 {}
507 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
508   { %only for the show command TEST!!
509     \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
510   }
511 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
512 ⟨/dvips⟩
```

dvipsvgm unneeded, or no-op

```
513 ⟨∗dvisvgm⟩
514 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1 {}
515 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
516   { %only for the show command TEST!!
517     \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
518   }
519 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
520 ⟨/dvisvgm⟩
```

(*End of definition for* `\__pdf_backend_PageResources_gput:nnn` *and* `\__pdf_backend_PageResources_-obj_gpush:`.)

1.9.1 Page resources /Properties + BDC operators

`\__pdf_backend_bdc:nn`
`\_pdf_backend_shipout_bdc:ee`
`\__pdf_backend_bdcobject:nn`
`\__pdf_backend_bdcobject:n`
`\__pdf_backend_bmc:n`
`\__pdf_backend_emc:`
`\__pdf_backend_PageResources_gpush:n`

`\__pdf_backend_bdc:nn`, `\__pdf_backend_shipout_bdc:ee`, `\__pdf_backend_bdcobject:nn`, `\__pdf_backend_bdcobject:n`, `\__pdf_backend_bmc:n` and `\__pdf_backend_emc:` are the backend command that create the bdc/emc marker and store the properties. `\__pdf_backend_PageResources_gpush:n` outputs the /Properties and/or the other resources for the current page.

```
521 % pdftex and luatex (and perhaps dvips ...) need to know if there are in a
522 % xform stream ...
523 ⟨∗drivers⟩
524 \bool_new:N \l__pdf_backend_xform_bool
525 ⟨/drivers⟩
526 ⟨∗dvips⟩
527 % dvips is easy: create an object, and reference it in the bdc
528 % ghostscript will then automatically replace it by a name
529 % and add the name to the /Properties dict
530 % special variant von accsupp
531 % https://chat.stackexchange.com/transcript/message/50831812#50831812
532 %
533 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2 % #1 eg. Span, #2: dict_content
534   {
535     \__pdf_backend_pdfmark:n{/#1~<<#2>>~/BDC}
536   }
537
538 \bool_if:NT\l__pdfmanagement_delayed_shipout_bool
539   {
540     \cs_new_protected:Npn \__pdf_backend_bdc_shipout:ee #1 #2 % #1 eg. Span, #2: dict_content
541       {
542         \__kernel_backend_shipout_literal:e
```

```
543            {ps: SDict ~ begin ~ mark /#1~<<#2>>~/BDC ~ pdfmark ~ end }
544        }
545    }
546
547 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
548    {
549      \__pdf_backend_pdfmark:e{/#1~\pdf_object_ref:n{#2}~/BDC}
550    }
551 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1  % #1 eg. Span,
552    {
553      \__pdf_backend_pdfmark:e{/#1~\__pdf_backend_object_last:~/BDC}
554    }
555 \cs_set_protected:Npn \__pdf_backend_emc:
556    {
557      \__pdf_backend_pdfmark:n{/EMC} %
558    }
559 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
560    {
561      \__pdf_backend_pdfmark:n{/#1~/BMC} %
562    }
563 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
564
565 ⟨/dvips⟩
566 ⟨*dvisvgm⟩
567 % dvisvgm should do nothing
568 %
569 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2 % #1 eg. Span, #2: dict_content
570    {}
571 \bool_if:NT\l__pdfmanagement_delayed_shipout_bool
572    {
573      \cs_set_protected:Npn \__pdf_backend_shipout_bdc:ee #1 #2 % #1 eg. Span, #2: dict_conten
574        {}
575    }
576 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
577    {}
578 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1  % #1 eg. Span,
579    {}
580 \cs_set_protected:Npn \__pdf_backend_emc:
581    {}
582 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
583    {}
584 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
585
586 ⟨/dvisvgm⟩
587
588 % xetex has to create the entries in the /Properties manually
589 % (like the other backends)
590 % use pdfbase special
591 % https://chat.stackexchange.com/transcript/message/50832016#50832016
592 % the property is added to xform resources automatically,
593 % no need to worry about it.
594 ⟨*dvipdfmx | xdvipdfmx⟩
595  \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
596     {
```

```
597     \int_gincr:N \g__pdf_backend_name_int
598     \__kernel_backend_literal:e
599       {
600         pdf:code~/#1/l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC
601       }
602     \__kernel_backend_literal:e
603       {
604         pdf:put~@resources~
605           <<
606             /Properties~
607               <<
608                 /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl
609                 \pdf_object_ref:n { #2 }
610               >>
611           >>
612       }
613   }
614 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1  % #1 eg. Span
615   {
616     \int_gincr:N \g__pdf_backend_name_int
617     \__kernel_backend_literal:e
618       {
619         pdf:code~/\exp_not:n{#1}/l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC
620       }
621     \__kernel_backend_literal:e
622       {
623         pdf:put~@resources~
624           <<
625             /Properties~
626               <<
627                 /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl
628                 \__pdf_backend_object_last:
629               >>
630           >>
631       }
632   }
633 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
634   {
635     \__kernel_backend_literal:n {pdf:code~/#1~BMC}  %pdfbase
636   }
637
638 %this require management
639 \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
640   {
641     \pdf_object_unnamed_write:nn { dict }{ #2 }
642     \__pdf_backend_bdcobject:n { #1 }
643   }
644
645 \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
646   {
647     \__kernel_backend_literal:n {pdf:code~ /#1~<<#2>>~BDC }
648   }
649
650 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2
```

```
651     {
652       \bool_if:NTF  \g__pdfmanagement_active_bool
653         {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contobj:nn}
654         {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn}
655         \__pdf_backend_bdc:nn {#1}{#2}
656     }
657
658   \bool_if:NT\l__pdfmanagement_delayed_shipout_bool
659     {
660       \cs_set_protected:Npn \__pdf_backend_bdc_shipout_contstream:ee #1 #2
661         {
662           \__kernel_backend_shipout_literal:e {pdf:code~ /#1~<<#2>>~BDC }
663         }
664       \cs_set_eq:NN \__pdf_backend_bdc_shipout:ee \__pdf_backend_bdc_shipout_contstream:ee
665     }
666   \cs_set_protected:Npn \__pdf_backend_emc:
667     {
668       \__kernel_backend_literal:n {pdf:code~EMC}  %pdfbase
669     }
670     % properties are handled automatically, but the other resources should be added
671     % at shipout
672   \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1
673     {
674       \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
675         {
676           \prop_if_empty:cF { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/##1} }
677             {
678               \__kernel_backend_literal:e
679                 {
680                   pdf:put~@resources~
681                     <</##1~\pdf_object_ref:n {__pdf/Page/Resources/##1}>>
682                 }
683             }
684         }
685     }
686 ⟨/dvipdfmx | xdvipdfmx⟩
687 % luatex + pdftex
688 ⟨∗luatex⟩
689 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
690     {
691       \int_gincr:N \g__pdf_backend_name_int
692       \__kernel_backend_literal_page:e
693         { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
694       \bool_if:NTF \l__pdf_backend_xform_bool
695         {
696           \pdfdict_gput:nee
697             { g__pdf_Core/Xform/Resources/Properties }
698             { l3pdf\int_use:N\g__pdf_backend_name_int }
699             { \pdf_object_ref:n { #2 } }
700         }
701         {
702           \exp_args:Ne \tex_latelua:D
703             {
704               ltx.pdf.Page_Resources_Properties_gput
```

16

```
705              (
706                tex.count["g_shipout_readonly_int"],
707                "l3pdf\int_use:N\g__pdf_backend_name_int",
708                "\pdf_object_ref:n { #2 }"
709              )
710            }
711        }
712    }
713 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1% #1 eg. Span
714    {
715      \int_gincr:N \g__pdf_backend_name_int
716      \__kernel_backend_literal_page:e
717        { /\exp_not:n{#1} ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
718      \bool_if:NTF \l__pdf_backend_xform_bool
719        {
720          \pdfdict_gput:nee %no handler needed
721            { g__pdf_Core/Xform/Resources/Properties }
722            { l3pdf\int_use:N\g__pdf_backend_name_int }
723            { \__pdf_backend_object_last: }
724        }
725        {
726          \exp_args:Ne \tex_latelua:D
727            {
728              ltx.pdf.Page_Resources_Properties_gput
729                (
730                  tex.count["g_shipout_readonly_int"],
731                  "l3pdf\int_use:N\g__pdf_backend_name_int",
732                  "\__pdf_backend_object_last:"
733                )
734            }
735        }
736    }
737 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
738    {
739      \__kernel_backend_literal_page:n { /#1~BMC }
740    }
741 \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
742    {
743      \pdf_object_unnamed_write:nn { dict } { #2 }
744      \__pdf_backend_bdcobject:n { #1 }
745    }
746 \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
747    {
748      \__kernel_backend_literal_page:n { /#1~<<#2>>~BDC }
749    }
750
751 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2
752    {
753      \bool_if:NTF  \g__pdfmanagement_active_bool
754        {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contobj:nn}
755        {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn}
756      \__pdf_backend_bdc:nn {#1}{#2}
757    }
758
```

```
759 \bool_if:NT\l__pdfmanagement_delayed_shipout_bool
760   {
761     \cs_set_protected:Npn \__pdf_backend_bdc_shipout_contstream:ee #1 #2
762       {
763         \__kernel_backend_shipout_literal_page:e { /#1~<<#2>>~BDC }
764       }
765     \cs_set_eq:NN \__pdf_backend_bdc_shipout:ee \__pdf_backend_bdc_shipout_contstream:ee
766   }
767
768 \cs_set_protected:Npn \__pdf_backend_emc:
769   {
770     \__kernel_backend_literal_page:n { EMC }
771   }
772
773 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
774 ⟨/luatex⟩
775 ⟨*pdftex⟩
776 % pdflatex is the most complicated as it has to go through the aux ...
777 % the push command is extended to take other resources too
778 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
779   {
780     \int_gincr:N \g__pdf_backend_name_int
781     \__kernel_backend_literal_page:e
782       { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
783     % code to set the property ....
784     \int_gincr:N\g__pdf_backend_resourceid_int
785     \bool_if:NTF \l__pdf_backend_xform_bool
786       {
787         \pdfdict_gput:nee %no handler needed
788           { g__pdf_Core/Xform/Resources/Properties }
789           { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
790           { \pdf_object_ref:n { #2 } }
791       }
792       {
793         \__pdf_backend_record_abspage:e {l3pdf\int_use:N\g__pdf_backend_resourceid_int}
794         \tl_set:Ne \l__pdf_tmpa_tl
795           {
796             \__pdf_backend_ref_abspage:e{l3pdf\int_use:N\g__pdf_backend_resourceid_int}
797           }
798         \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties
799           {
800             \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
801           }
802         \pdfdict_gput:nee
803           { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
804           { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
805           { \pdf_object_ref:n{#2} }
806       }
807   }
808 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1% #1 eg. Span
809   {
810     \int_gincr:N \g__pdf_backend_name_int
811     \__kernel_backend_literal_page:e
812       { /\exp_not:n{#1} ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
```

```
813    % code to set the property ....
814    \int_gincr:N\g__pdf_backend_resourceid_int
815    \bool_if:NTF \l__pdf_backend_xform_bool
816      {
817        \pdfdict_gput:nee
818          { g__pdf_Core/Xform/Resources/Properties }
819          { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
820          { \__pdf_backend_object_last: }
821      }
822      {
823        \__pdf_backend_record_abspage:e{l3pdf\int_use:N\g__pdf_backend_resourceid_int}
824        \tl_set:Ne \l__pdf_tmpa_tl
825          {
826            \__pdf_backend_ref_abspage:e{l3pdf\int_use:N\g__pdf_backend_resourceid_int}
827          }
828        \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties
829          {
830            \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
831          }
832        \pdfdict_gput:nee
833          { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
834          { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
835          { \__pdf_backend_object_last: }
836        %\pdfdict_show:n { g_backend_Page\l__pdf_tmpa_tl/Resources/Properties }
837      }
838  }
839  \cs_set_protected:Npn \__pdf_backend_bmc:n #1
840    {
841      \__kernel_backend_literal_page:n { /#1~BMC }
842    }
843  \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
844    {
845      \pdf_object_unnamed_write:nn { dict } { #2 }
846      \__pdf_backend_bdcobject:n { #1 }
847    }
848  \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
849    {
850      \__kernel_backend_literal_page:n { /#1~<<#2>>~BDC }
851    }
852
853  \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2
854    {
855      \bool_if:NTF  \g__pdfmanagement_active_bool
856        {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contobj:nn}
857        {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn}
858      \__pdf_backend_bdc:nn {#1}{#2}
859    }
860  \bool_if:NT\l__pdfmanagement_delayed_shipout_bool
861    {
862      \cs_set_protected:Npn \__pdf_backend_bdc_shipout_contstream:ee #1 #2
863          {
864            \__kernel_backend_shipout_literal_page:e { /#1~<<#2>>~BDC }
865          }
866      \cs_set_eq:NN \__pdf_backend_bdc_shipout:ee \__pdf_backend_bdc_shipout_contstream:ee
```

19

```
867    }
868
869  \cs_set_protected:Npn \__pdf_backend_emc:
870    {
871      \__kernel_backend_literal_page:n { EMC }
872    }
873
874  \cs_new:Npn \__pdf_backend_PageResources_gpush_aux:n #1 %#1 ExtGState etc
875    {
876      \prop_if_empty:cF
877        { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/#1} }
878        {
879          \pdfdict_item:ne { #1 }{ \pdf_object_ref:n {__pdf/Page/Resources/#1}}
880        }
881    }
882
883  \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1
884    {
885      \exp_args:NNe \tex_global:D \tex_pdfpageresources:D
886        {
887          \prop_if_exist:cT
888            { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1/Resources/Properties } }
889            {
890              /Properties~
891                <<
892                  \prop_map_function:cN
893                    { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1/Resources/Propert
894                    \pdfdict_item:ne
895                >>
896            }
897          %% add ExtGState etc
898          \clist_map_function:NN
899            \c__pdf_backend_PageResources_clist
900            \__pdf_backend_PageResources_gpush_aux:n
901        }
902    }
903
904  ⟨/pdftex⟩
```

(*End of definition for* `\__pdf_backend_bdc:nn` *and others.*)

1.10 "Catalog" & subdirectories (pdfcatalog)

The backend command is already in the driver: `\__pdf_backend_catalog_gput:nn`

1.10.1 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.

```
905  % pdflatex
906  ⟨*pdftex⟩
```

```
907 \cs_new_protected:Npn \__pdf_backend_Names_gpush:nn #1 #2 %#1 name of name tree, #2 array co
908   {
909       \pdf_object_unnamed_write:nn {dict} {/Names [#2] }
910       \tex_pdfnames:D {/#1~\pdf_object_ref_last:}
911   }
912 ⟨/pdftex⟩
913 ⟨∗luatex⟩
914 \cs_new_protected:Npn \__pdf_backend_Names_gpush:nn #1 #2 %#1 name of name tree, #2 array co
915   {
916       \pdf_object_unnamed_write:nn {dict} {/Names [#2] }
917       \tex_pdfextension:D~names~ {/#1~\pdf_object_ref_last:}
918   }
919 ⟨/luatex⟩
920 ⟨∗dvipdfmx | xdvipdfmx⟩
921 \cs_new_protected:Npn \__pdf_backend_Names_gpush:nn #1 #2 %#1 name of name tree, #2 array co
922   {
923       \pdf_object_unnamed_write:nn {dict} {/Names [#2] }
924       \__pdf_backend:e {put~@names~<</#1~\pdf_object_ref_last: >>}
925   }
926 ⟨/dvipdfmx | xdvipdfmx⟩
927
928 %dvips: noop
929 ⟨∗dvips⟩
930 \cs_new_protected:Npn \__pdf_backend_Names_gpush:nn #1 #2  {}
931 ⟨/dvips⟩
932 %dvisvgm: noop
933 ⟨∗dvisvgm⟩
934 \cs_new_protected:Npn \__pdf_backend_Names_gpush:nn #1 #2 {}
935 ⟨/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.

```
936 ⟨∗pdftex | luatex | dvipdfmx | xdvipdfmx⟩
937 \cs_new_protected:Npn  \__pdf_backend_NamesEmbeddedFiles_add:nn #1 #2 {}
938 ⟨/pdftex | luatex | dvipdfmx | xdvipdfmx⟩
939 ⟨∗dvips⟩
940 \cs_new_protected:Npn  \__pdf_backend_NamesEmbeddedFiles_add:nn #1 #2
941       {
942           \__pdf_backend_pdfmark:e
943             {
944                 /Name~#1~
945                 /FS~#2~
946                 /EMBED
947             }
948       }
949 ⟨/dvips⟩
950 ⟨∗dvisvgm⟩
951 %no op. Or is there any sensible use for it?
952 \cs_new_protected:Npn  \__pdf_backend_NamesEmbeddedFiles_add:nn #1 #2
953       {}
954
955 ⟨/dvisvgm⟩
```

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

1.10.2 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.

```
956 ⟨∗drivers⟩
957 \cs_new_protected:Npn \__pdf_backend_link_off:{}
958 \cs_new_protected:Npn \__pdf_backend_link_on: {}
959 ⟨/drivers⟩
960 ⟨∗pdftex⟩
961 \cs_if_exist:NT \pdfrunninglinkoff
962  {
963    \cs_set_protected:Npn \__pdf_backend_link_off:
964      {
965        \pdfrunninglinkoff
966      }
967    \cs_set_protected:Npn \__pdf_backend_link_on:
968      {
969        \pdfrunninglinkon
970      }
971  }
972 ⟨/pdftex⟩
973 ⟨∗luatex⟩
974 \int_compare:nNnT {\tex_luatexversion:D } > {112}
975  {
976    \cs_set_protected:Npn \__pdf_backend_link_off:
977      {
978        \pdfextension linkstate 1
979      }
980    \cs_set_protected:Npn \__pdf_backend_link_on:
981      {
982        \pdfextension linkstate 0
983      }
984  }
985 ⟨/luatex⟩
986 ⟨∗dvipdfmx | xdvipdfmx⟩
987   \cs_set_protected:Npn \__pdf_backend_link_off:
988      {
989        \__pdf_backend:n { nolink }
990      }
991    \cs_set_protected:Npn \__pdf_backend_link_on:
992      {
993        \__pdf_backend:n { link }
994      }
995 ⟨/dvipdfmx | xdvipdfmx⟩
```

1.10.3 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
__pdf_backend_xform_ref:n

```
996 ⟨∗pdftex⟩
997 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
998 % #1 name
999 % #2 attributes
1000 % #3 resources
1001 % #4 content, not necessarily a box!
1002   {
1003     \hbox_set:Nn \l__pdf_backend_tmpa_box
1004       {
1005         \bool_set_true:N \l__pdf_backend_xform_bool
1006         \prop_gclear:c {\__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties }}
1007         #4
1008       }
1009     %store the dimensions
1010     \tl_const:ce
1011       { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1012       { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1013     \tl_const:ce
1014       { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1015       { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1016     \tl_const:ce
1017       { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1018       { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1019     %% do we need to test if #2 and #3 are empty??
1020     \tex_immediate:D \tex_pdfxform:D
1021       ~ attr      ~ { #2 }
1022     %% which other resources should be default? Is an argument actually needed?
1023       ~ resources ~
1024       {
1025         #3
1026         \int_compare:nNnT
1027           { \prop_count:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Propertie
1028           >
1029           { 0 }
1030           {
1031             /Properties~
1032               <<
1033                 \pdfdict_use:n { g__pdf_Core/Xform/Resources/Properties }
1034               >>
1035           }

1037         \prop_if_empty:cF
1038           { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
1039           {
1040             /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1041           }
1042         \prop_if_empty:cF
1043           { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
1044           {
1045             /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1046           }
```

23

```
1047        \prop_if_empty:cF
1048          { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Shading } }
1049          {
1050            /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1051          }
1052        \prop_if_empty:cF
1053          { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
1054          {
1055            /ColorSpace~ \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1056          }
1057      }
1058      \l__pdf_backend_tmpa_box
1059    \int_const:cn
1060      { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1061      { \tex_pdflastxform:D }
1062  }
1063
1064 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1065  {
1066    \tex_pdfrefxform:D
1067      \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1068      \scan_stop:
1069  }
1070
1071 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1072  {
1073    \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R
1074  }
1075 ⟨/pdftex⟩
1076 ⟨*luatex⟩
1077 %luatex
1078 %nearly identical but not completely ...
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    \hbox_set:Nn \l__pdf_backend_tmpa_box
1086      {
1087        \bool_set_true:N \l__pdf_backend_xform_bool
1088        \prop_gclear:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties }
1089        #4
1090      }
1091    \tl_const:ce
1092      { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1093      { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1094    \tl_const:ce
1095      { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1096      { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1097    \tl_const:ce
1098      { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1099      { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1100    %% do we need to test if #2 and #3 are empty??
```

```
1101     \tex_immediate:D \tex_pdfxform:D
1102       ~ attr       ~ { #2 }
1103       %% which resources should be default? Is an argument actually needed?
1104       ~  resources ~
1105       {
1106         #3
1107         \int_compare:nNnT
1108           {\prop_count:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties
1109           >
1110           { 0 }
1111           {
1112             /Properties~
1113               <<
1114                 \pdfdict_use:n { g__pdf_Core/Xform/Resources/Properties }
1115               >>
1116           }
1117         \prop_if_empty:cF
1118           { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
1119           {
1120             /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1121           }
1122         \prop_if_empty:cF
1123           { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
1124           {
1125             /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1126           }
1127         \prop_if_empty:cF
1128           { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Shading } }
1129           {
1130             /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1131           }
1132         \prop_if_empty:cF
1133           { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
1134           {
1135             /ColorSpace~ \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1136           }
1137       }
1138       \l__pdf_backend_tmpa_box
1139       \int_const:cn
1140         { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1141         { \tex_pdflastxform:D }
1142   }
1143
1144 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 %protected as with xelatex
1145   {
1146     \tex_pdfrefxform:D \int_use:c
1147       {
1148         c__pdf_backend_xform_ \tl_to_str:n {#1} _int
1149       }
1150     \scan_stop:
1151   }
1152
1153 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1154   { \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R }
```

```
1155
1156  ⟨/luatex⟩
1157  ⟨*dvipdfmx | xdvipdfmx⟩
1158  % xetex
1159     % it needs a bit testing if it really works to set the box to 0 before the special ...
1160     % does it disturb viewing the xobject?
1161     % what happens with the resources (bdc)? (should work as they are specials too)
1162     % xetex requires that the special is in horizontal mode. This means it affects
1163     % typesetting. But we can no delay the whole form code to shipout
1164     % as the object reference and the size is often wanted on the current page.
1165     % so we need to allocate a box - but probably they won't be thousands xform
1166     % in a document so it shouldn't matter.
1167     \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
1168     % #1 name
1169     % #2 attributes
1170     % #3 resources
1171     % #4 content, not necessarily a box!
1172       {
1173         \int_gincr:N \g__pdf_backend_object_int
1174         \int_const:cn
1175           { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1176           { \g__pdf_backend_object_int }
1177         \box_new:c { g__pdf_backend_xform_#1_box }
1178         \hbox_gset:cn { g__pdf_backend_xform_#1_box }
1179           {
1180             \bool_set_true:N \l__pdf_backend_xform_bool
1181             #4
1182           }
1183         \tl_const:ce
1184           { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1185           { \tex_the:D \box_wd:c { g__pdf_backend_xform_#1_box } }
1186         \tl_const:ce
1187           { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1188           { \tex_the:D \box_ht:c { g__pdf_backend_xform_#1_box } }
1189         \tl_const:ce
1190           { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1191           { \tex_the:D \box_dp:c { g__pdf_backend_xform_#1_box } }
1192         \box_set_dp:cn  { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1193         \box_set_ht:cn  { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1194         \box_set_wd:cn  { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1195         \hook_gput_next_code:nn {shipout/background}
1196           {
1197             \mode_leave_vertical: %needed, the xform disappears without it.
1198             \__pdf_backend:e
1199               {
1200                 bxobj  ~ \__pdf_backend_xform_ref:n  { #1 }
1201                 \c_space_tl width  ~ \pdfxform_wd:n { #1 }
1202                 \c_space_tl height ~ \pdfxform_ht:n { #1 }
1203                 \c_space_tl depth  ~ \pdfxform_dp:n { #1 }
1204               }
1205             \box_use_drop:c { g__pdf_backend_xform_#1_box }
1206             \__pdf_backend:e {put ~ @resources ~<<#3>> }
1207             \__pdf_backend:e
1208               {
```

26

```
1209                    put~ @resources ~
1210                      <<
1211                        /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1212                      >>
1213                  }
1214              \__pdf_backend:e
1215                  {
1216                    put~ @resources ~
1217                    <<
1218                      /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1219                    >>
1220                  }
1221              \__pdf_backend:e
1222                  {
1223                    put~ @resources ~
1224                    <<
1225                      /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1226                    >>
1227                  }
1228              \__pdf_backend:e
1229                  {
1230                    put~ @resources ~
1231                    <<
1232                      /ColorSpace~
1233                      \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1234                    >>
1235                  }
1236              \__pdf_backend:e {exobj ~<<#2>>}
1237          }
1238      }
1239
1240
1241
1242    \cs_new:Npn \__pdf_backend_xform_ref:n #1
1243      {
1244        @pdf.xform \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1245      }
1246
1247    \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1248      {
1249        \hbox_set:Nn \l__pdf_backend_tmpa_box
1250          {
1251            \__pdf_backend:e
1252              {
1253                uxobj~ \__pdf_backend_xform_ref:n { #1 }
1254              }
1255          }
1256        \box_set_wd:Nn  \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1257        \box_set_ht:Nn  \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1258        \box_set_dp:Nn  \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1259        \box_use_drop:N \l__pdf_backend_tmpa_box
1260      }
1261 ⟨/dvipdfmx | xdvipdfmx⟩
1262 ⟨∗dvisvgm⟩
```

```
1263  % unclear what it should do!!
1264  \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4 {}
1265  \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 {}
1266  \cs_new:Npn \__pdf_backend_xform_ref:n {}
1267  ⟨/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

```
1268  ⟨*dvips⟩
1269  \tl_new:N \l__pdf_backend_xform_tmpwd_tl
1270  \tl_new:N \l__pdf_backend_xform_tmpdp_tl
1271  \tl_new:N \l__pdf_backend_xform_tmpht_tl

1272  \cs_new_protected:Npn\__pdf_backend_xform_new:nnnn #1 #2 #3 #4 % #1 name, #2 attribute, #4
1273    {
1274      \int_gincr:N \g__pdf_backend_object_int
1275      \int_const:cn
1276        { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1277        { \g__pdf_backend_object_int }
1278
1279      \hbox_set:Nn \l__pdf_backend_tmpa_box
1280        {
1281          \bool_set_true:N \l__pdf_backend_xform_bool
1282          \prop_gclear:c {\__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties }}
1283          #4
1284        }
1285    %store the dimensions
1286      \tl_const:ce
1287        { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1288        { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1289      \tl_const:ce
1290        { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1291        { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1292      \tl_const:ce
1293        { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1294        { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1295    %store content dimensions in DPI units (Dots) (code from issue 25)
1296      \tl_set:Ne\l__pdf_backend_xform_tmpwd_tl
1297        {
1298          \dim_to_decimal_in_sp:n{ \box_wd:N \l__pdf_backend_tmpa_box }~
1299          65536~div~72.27~div~DVImag~mul~Resolution~mul~
1300        }
1301      \tl_set:Ne\l__pdf_backend_xform_tmpht_tl
1302        {
1303          \dim_to_decimal_in_sp:n{ \box_ht:N \l__pdf_backend_tmpa_box }~
1304          65536~div~72.27~div~DVImag~mul~VResolution~mul~
1305        }
1306      \tl_set:Ne\l__pdf_backend_xform_tmpdp_tl
1307        {
1308          \dim_to_decimal_in_sp:n{ \box_dp:N \l__pdf_backend_tmpa_box }~
1309          65536~div~72.27~div~DVImag~mul~VResolution~mul~
1310        }
1311      % mirror the box
```

```
1312    %\box_scale:Nnn \l__pdf_backend_tmpa_box {1} {-1}
1313    \hbox_set:Nn\l__pdf_backend_tmpb_box
1314      {
1315        \__kernel_backend_postscript:e
1316          {
1317            gsave~currentpoint~
1318            initclip~ % restore default clipping path (page device/whole page)
1319            clippath~pathbbox~newpath~pop~pop~
1320            \tl_use:N\l__pdf_backend_xform_tmpdp_tl~add~translate~
1321            mark~
1322              /_objdef~{ pdf.obj \int_use:N\g__pdf_backend_object_int }\c_space_tl~
1323              /BBox[
1324                0~
1325                \tl_use:N\l__pdf_backend_xform_tmpht_tl~
1326                \tl_use:N\l__pdf_backend_xform_tmpwd_tl~
1327                \tl_use:N\l__pdf_backend_xform_tmpdp_tl~
1328                neg
1329              ]
1330              \str_if_eq:eeF{#1}{}
1331              {
1332                product~(Distiller)~search~{pop~pop~pop~#2}{pop}ifelse~
1333              }
1334            /BP~pdfmark~1~-1~scale~neg~exch~neg~exch~translate
1335          }
1336        \box_use_drop:N\l__pdf_backend_tmpa_box
1337        \__kernel_backend_postscript:n
1338          {
1339            mark ~ /EP~pdfmark  ~ grestore
1340          }
1341        \str_if_eq:eeF{#1}{}
1342          {
1343            \__kernel_backend_postscript:e
1344              {
1345                product~(Ghostscript)~search~
1346                  {
1347                    pop~pop~pop~
1348                    mark~
1349                    { pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }
1350                      ~<<#2>>~/PUT~pdfmark
1351                  }{pop}ifelse
1352              }
1353          }
1354      }
1355    \box_set_dp:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1356    \box_set_ht:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1357    \box_set_wd:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1358    \hook_gput_code:nnn {begindocument/end}{pdfxform}
1359      {
1360        \mode_leave_vertical:
1361        \box_use:N\l__pdf_backend_tmpb_box
1362      }
1363  }
1364
1365
```

```
1366  \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1367    {
1368      \hbox_set:Nn \l__pdf_backend_tmpa_box
1369        {
1370          \__kernel_backend_postscript:e
1371            {
1372              gsave~currentpoint~translate~1~-1~scale~
1373              mark~{ pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int }}~
1374              /SP~pdfmark ~ grestore
1375            }
1376        }
1377      \box_set_wd:Nn  \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1378      \box_set_ht:Nn  \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1379      \box_set_dp:Nn  \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1380      \box_use_drop:N \l__pdf_backend_tmpa_box
1381    }
1382  \cs_new:Npn \__pdf_backend_xform_ref:n #1
1383    {
1384      { pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }
1385    }
1386
1387  ⟨/dvips⟩
1388  ⟨*drivers⟩
1389  %% all
1390  \prg_new_conditional:Npnn \__pdf_backend_xform_if_exist:n #1 { p , T , F , TF }
1391    {
1392      \int_if_exist:cTF { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1393        { \prg_return_true: }
1394        { \prg_return_false:}
1395    }
1396  \prg_new_eq_conditional:NNn \pdfxform_if_exist:n\__pdf_backend_xform_if_exist:n
1397    { TF , T , F , p }
1398  ⟨/drivers⟩
```

(*End of definition for* __pdf_backend_xform_new:nnnn, __pdf_backend_xform_use:n, *and* __pdf_-
backend_xform_ref:n.)

1.11   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 TeXlive 2022

(earlier in miktex) both engine will knew 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.

The needed code differ depending on if structure objects use standard or indexed object names. At the end we will probably always use indexed objects, but for now we offer both options.

`\l_pdf_current_structure_destination_tl` This command holds the name of the structure object to use in the following commands 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
```

or if indexed structure object names are used

```
\tl_set:Nn \l_pdf_current_structure_destination_tl { {__tag/struct}{\g__tag_struct_sta
```

```
1399 ⟨*drivers⟩
1400 \tl_new:N    \l_pdf_current_structure_destination_tl
1401 ⟨/drivers⟩
```

(*End of 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
\__pdf_backend_destination:nn       -> \__pdf_backend_indexed_structure_destination:nn
\__pdf_backend_destination:nnnn -> \__pdf_backend_indexed_structure_destination:nnnn
\__pdf_backend_link_begin_goto:nnw  -> \__pdf_backend_indexed_link_begin_structure_goto
```

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:`
`\pdf_activate_indexed_structure_destination:`

```
1402 ⟨*drivers⟩
1403 \cs_new_protected:Npn \pdf_activate_structure_destination:
1404  {
1405    \cs_gset_eq:NN \__pdf_backend_destination:nn       \__pdf_backend_structure_destination:n
1406    \cs_gset_eq:NN \__pdf_backend_destination:nnnn     \__pdf_backend_structure_destination:n
1407    \cs_gset_eq:NN \__pdf_backend_link_begin_goto:nnw \__pdf_backend_link_begin_structure_go
1408  }
1409 \cs_new_protected:Npn \pdf_activate_indexed_structure_destination:
1410  {
1411    \cs_gset_eq:NN \__pdf_backend_destination:nn       \__pdf_backend_indexed_structure_desti
1412    \cs_gset_eq:NN \__pdf_backend_destination:nnnn     \__pdf_backend_indexed_structure_desti
1413    \cs_gset_eq:NN \__pdf_backend_link_begin_goto:nnw \__pdf_backend_link_begin_structure_go
1414  }
1415 ⟨/drivers⟩
```

31

(*End of definition for* `\pdf_activate_structure_destination:` *and* `\pdf_activate_indexed_structure_-`
`destination:`*. These functions are documented on page* **??**.)

Now the driver dependent 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.

```
1416 ⟨*drivers⟩
1417 \cs_set_eq:NN \__pdf_backend_structure_destination:nn        \__pdf_backend_destination:nn
1418 \cs_set_eq:NN \__pdf_backend_structure_destination:nnnn      \__pdf_backend_destination:nnnn
1419 \cs_set_eq:NN \__pdf_backend_link_begin_structure_goto:nnw  \__pdf_backend_link_begin_goto:n
1420 \cs_set_eq:NN \__pdf_backend_indexed_structure_destination:nn        \__pdf_backend_destinati
1421 \cs_set_eq:NN \__pdf_backend_indexed_structure_destination:nnnn      \__pdf_backend_destinati
1422 ⟨/drivers⟩
```

__pdf_backend_structure_destination:nn
__pdf_backend_structure_destination:nnnn
__pdf_backend_link_begin_structure_goto:nnw

These commands are the backend commands to create a destination. which 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:

```
1423 ⟨*xdvipdfmx | dvipdfmx⟩
1424 \cs_set_protected:Npn \__pdf_backend_structure_destination:nn #1#2
1425   {
1426     \__pdf_backend:e
1427       {
1428         dest ~ ( \exp_not:n {#1} )
1429         [
1430           @thispage
1431           \str_case:nnF {#2}
1432             {
1433               { xyz }   { /XYZ ~ @xpos ~ @ypos ~ null }
1434               { fit }   { /Fit }
1435               { fitb } { /FitB }
1436               { fitbh } { /FitBH }
1437               { fitbv } { /FitBV ~ @xpos }
1438               { fith } { /FitH ~ @ypos }
1439               { fitv } { /FitV ~ @xpos }
1440               { fitr } { /Fit }
1441             }
1442           { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1443         ]
1444       }
```

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.

```
1445     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1446       {
1447         \__pdf_backend:e
1448           {
1449             obj ~ @pdf.SDest.\exp_not:n{#1}
1450             [
1451               \exp_args:Ne \pdf_object_ref:n { \l_pdf_current_structure_destination_tl }
1452               \str_case:nnF {#2}
1453                 {
1454                   { xyz }   { /XYZ ~ @xpos ~ @ypos ~ null }
1455                   { fit }   { /Fit }
```

```
1456                { fitb }  { /FitB }
1457                { fitbh } { /FitBH }
1458                { fitbv } { /FitBV ~ @xpos }
1459                { fith }  { /FitH ~ @ypos }
1460                { fitv }  { /FitV ~ @xpos }
1461                { fitr }  { /Fit }
1462              }
1463              { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1464          ]
1465        }
1466      }
1467  }
```

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

```
1468  \cs_new_protected:Npn \__pdf_backend_structure_destination_aux:nnnn #1#2#3#4
1469    {
1470      \vbox_to_zero:n
1471        {
1472          \__kernel_kern:n {#4}
1473          \hbox:n
1474            {
1475              \__pdf_backend:n { obj ~ @pdf_ #2 _llx ~ @xpos }
1476              \__pdf_backend:n { obj ~ @pdf_ #2 _lly ~ @ypos }
1477            }
1478          \tex_vss:D
1479        }
1480      \__kernel_kern:n {#1}
1481      \vbox_to_zero:n
1482        {
1483          \__kernel_kern:n { -#3 }
1484          \hbox:n
1485            {
1486              \__pdf_backend:n
1487                {
1488                  dest ~ (#2)
1489                  [
1490                    @thispage
1491                    /FitR ~
1492                      @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1493                      @xpos ~ @ypos
1494                  ]
1495                }
```

Here we add the structure destination to the same box

```
1496              \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1497                {
1498                  \__pdf_backend:e
1499                    {
1500                      obj ~ @pdf.SDest.\exp_not:n{#2}
1501                      [
1502                        \exp_args:Ne \pdf_object_ref:n { \l_pdf_current_structure_destination_
1503                        /FitR ~
1504                          @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1505                          @xpos ~ @ypos
```

```
1506                               ]
1507                             }
1508                           }
1509                         }
1510                       \tex_vss:D
1511                     }
1512                 \__kernel_kern:n { -#1 }
1513             }
```

And now we redefine the destination command:

```
1514 \cs_set_protected:Npn \__pdf_backend_structure_destination:nnnn #1#2#3#4
1515   {
1516     \exp_args:Ne \__pdf_backend_structure_destination_aux:nnnn
1517       { \dim_eval:n {#2} } {#1} {#3} {#4}
1518   }
```

At last the goto link.

```
1519 \cs_set_protected:Npn \__pdf_backend_link_begin_structure_goto:nnw #1#2
1520   {
1521     \__pdf_backend_link_begin:n { #1 /Subtype /Link /A << /S /GoTo /D ( #2 ) /SD~@pdf.SDest.
1522   }
1523 ⟨/xdvipdfmx | dvipdfmx⟩
```

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

```
1524 ⟨*pdftex⟩
1525 \bool_lazy_and:nnT
1526   { \int_compare_p:nNn {\tex_pdftexversion:D } > {139} }
1527   { \int_compare_p:nNn {\tex_pdftexrevision:D } > {23} }
1528   {
1529     \cs_set_protected:Npn \__pdf_backend_structure_destination:nn #1#2
1530       {
1531         \tex_pdfdest:D
1532           name {#1}
1533           \str_case:nnF {#2}
1534             {
1535               { xyz }   { xyz }
1536               { fit }   { fit }
1537               { fitb }  { fitb }
1538               { fitbh } { fitbh }
1539               { fitbv } { fitbv }
1540               { fith }  { fith }
1541               { fitv }  { fitv }
1542               { fitr }  { fitr }
1543             }
1544             { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1545           \scan_stop:
1546         \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1547           {
1548             \tex_pdfdest:D
1549               struct~
1550               \int_use:c
1551                 { c__pdf_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structure_destin
1552               name {#1}
1553               \str_case:nnF {#2}
1554                 {
```

```
1555                      { xyz }   { xyz }
1556                      { fit }   { fit }
1557                      { fitb }  { fitb }
1558                      { fitbh } { fitbh }
1559                      { fitbv } { fitbv }
1560                      { fith }  { fith }
1561                      { fitv }  { fitv }
1562                      { fitr }  { fitr }
1563                    }
1564                    { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1565                  \scan_stop:
1566              }
1567          }
1568      \cs_set_protected:Npn \__pdf_backend_structure_destination:nnnn #1#2#3#4
1569        {
1570          \tex_pdfdest:D
1571            name {#1}
1572            fitr ~
1573            width  \dim_eval:n {#2} ~
1574            height \dim_eval:n {#3} ~
1575            depth  \dim_eval:n {#4} \scan_stop:
1576          \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1577            {
1578              \tex_pdfdest:D
1579                struct~
1580                \int_use:c
1581                  { c__pdf_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structure_destinat
1582                name {#1}
1583                fitr ~
1584                width  \dim_eval:n {#2} ~
1585                height \dim_eval:n {#3} ~
1586                depth  \dim_eval:n {#4} \scan_stop:
1587            }
1588        }
1589      \cs_set_protected:Npn \__pdf_backend_link_begin_structure_goto:nnw #1#2
1590        {
1591          \__pdf_backend_link_begin:nnnw {#1} { goto~struct~name~{#2}~name } {#2}
1592        }
1593    }
1594 ⟨/pdftex⟩
```

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

```
1595 ⟨*luatex⟩
1596  \int_compare:nNnT {\directlua{tex.print(status.list()["development_id"])} } > {7468}
1597    {
1598      \cs_set_protected:Npn \__pdf_backend_structure_destination:nn #1#2
1599        {
1600          \tex_pdfextension:D dest
1601            name {#1}
1602            \str_case:nnF {#2}
1603              {
1604                { xyz }   { xyz }
1605                { fit }   { fit }
1606                { fitb }  { fitb }
1607                { fitbh } { fitbh }
```

```
1608              { fitbv }  { fitbv }
1609              { fith }   { fith }
1610              { fitv }   { fitv }
1611              { fitr }   { fitr }
1612            }
1613            { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1614          \scan_stop:
1615       \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1616         {
1617           \tex_pdfextension:D dest
1618             struct~
1619             \int_use:c
1620               { c__pdf_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structure_destin
1621             name {#1}
1622             \str_case:nnF {#2}
1623               {
1624                 { xyz }   { xyz }
1625                 { fit }   { fit }
1626                 { fitb }  { fitb }
1627                 { fitbh } { fitbh }
1628                 { fitbv } { fitbv }
1629                 { fith }  { fith }
1630                 { fitv }  { fitv }
1631                 { fitr }  { fitr }
1632               }
1633               { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1634             \scan_stop:
1635         }
1636       }
1637     \cs_set_protected:Npn \__pdf_backend_structure_destination:nnnn #1#2#3#4
1638       {
1639         \tex_pdfextension:D dest
1640          name {#1}
1641          fitr ~
1642          width  \dim_eval:n {#2} ~
1643          height \dim_eval:n {#3} ~
1644          depth  \dim_eval:n {#4} \scan_stop:
1645         \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1646           {
1647             \tex_pdfextension:D dest
1648               struct~
1649               \int_use:c
1650                 { c__pdf_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structure_destinat
1651               name {#1}
1652               fitr ~
1653               width  \dim_eval:n {#2} ~
1654               height \dim_eval:n {#3} ~
1655               depth  \dim_eval:n {#4} \scan_stop:
1656           }
1657       }
1658     \cs_set_protected:Npn \__pdf_backend_link_begin_structure_goto:nnw #1#2
1659       {
1660         \__pdf_backend_link_begin:nnnw {#1} { goto~struct~name~{#2}~name } {#2}
1661       }
```

```
1662     }
1663 ⟨/luatex⟩
```

(*End of definition for* `\__pdf_backend_structure_destination:nn`, `\__pdf_backend_structure_destination:nnnn`, *and* `\__pdf_backend_link_begin_structure_goto:nnw`.)

df_backend_indexed_structure_destination:nn
_backend_indexed_structure_destination:nnnn

This are the indexed variants of the commands to create a destination and 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:

```
1664 ⟨*xdvipdfmx | dvipdfmx⟩
1665 \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nn #1#2
1666   {
1667     \__pdf_backend:e
1668       {
1669         dest ~ ( \exp_not:n {#1} )
1670         [
1671           @thispage
1672           \str_case:nnF {#2}
1673             {
1674               { xyz }   { /XYZ ~ @xpos ~ @ypos ~ null }
1675               { fit }   { /Fit }
1676               { fitb }  { /FitB }
1677               { fitbh } { /FitBH }
1678               { fitbv } { /FitBV ~ @xpos }
1679               { fith }  { /FitH ~ @ypos }
1680               { fitv }  { /FitV ~ @xpos }
1681               { fitr }  { /Fit }
1682             }
1683             { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1684         ]
1685       }
```

We do not test anymore 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.

```
1686         \__pdf_backend:e
1687           {
1688             obj ~ @pdf.SDest.\exp_not:n{#1}
1689             [
1690               \exp_after:wN \pdf_object_ref_indexed:nn  \l_pdf_current_structure_destination_t
1691               \str_case:nnF {#2}
1692                 {
1693                   { xyz }   { /XYZ ~ @xpos ~ @ypos ~ null }
1694                   { fit }   { /Fit }
1695                   { fitb }  { /FitB }
1696                   { fitbh } { /FitBH }
1697                   { fitbv } { /FitBV ~ @xpos }
1698                   { fith }  { /FitH ~ @ypos }
1699                   { fitv }  { /FitV ~ @xpos }
1700                   { fitr }  { /Fit }
1701                 }
1702                 { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1703             ]
1704           }
1705   }
```

37

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

```
1706 \cs_new_protected:Npn \__pdf_backend_indexed_structure_destination_aux:nnnn #1#2#3#4
1707   {
1708     \vbox_to_zero:n
1709       {
1710         \__kernel_kern:n {#4}
1711         \hbox:n
1712           {
1713             \__pdf_backend:n { obj ~ @pdf_ #2 _llx ~ @xpos }
1714             \__pdf_backend:n { obj ~ @pdf_ #2 _lly ~ @ypos }
1715           }
1716         \tex_vss:D
1717       }
1718     \__kernel_kern:n {#1}
1719     \vbox_to_zero:n
1720       {
1721         \__kernel_kern:n { -#3 }
1722         \hbox:n
1723           {
1724             \__pdf_backend:n
1725               {
1726                 dest ~ (#2)
1727                 [
1728                   @thispage
1729                   /FitR ~
1730                     @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1731                     @xpos ~ @ypos
1732                 ]
1733               }
```

Here we add the structure destination to the same box

```
1734             \__pdf_backend:e
1735               {
1736                 obj ~ @pdf.SDest.\exp_not:n{#2}
1737                 [
1738                   \exp_after:wN \pdf_object_ref_indexed:nn  \l_pdf_current_structure_destin
1739                   /FitR ~
1740                     @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1741                     @xpos ~ @ypos
1742                 ]
1743               }
1744           }
1745         \tex_vss:D
1746       }
1747     \__kernel_kern:n { -#1 }
1748   }
```

And now we redefine the destination command:

```
1749 \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nnnn #1#2#3#4
1750   {
1751     \exp_args:Ne \__pdf_backend_indexed_structure_destination_aux:nnnn
1752       { \dim_eval:n {#2} } {#1} {#3} {#4}
1753   }
1754 ⟨/xdvipdfmx | dvipdfmx⟩
```

38

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

```
1755 ⟨∗pdftex⟩
1756 \bool_lazy_and:nnT
1757   { \int_compare_p:nNn {\tex_pdftexversion:D } > {139} }
1758   { \int_compare_p:nNn {\tex_pdftexrevision:D } > {23} }
1759   {
1760     \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nn #1#2
1761       {
1762         \tex_pdfdest:D
1763           name {#1}
1764           \str_case:nnF {#2}
1765             {
1766               { xyz }   { xyz }
1767               { fit }   { fit }
1768               { fitb }  { fitb }
1769               { fitbh } { fitbh }
1770               { fitbv } { fitbv }
1771               { fith }  { fith }
1772               { fitv }  { fitv }
1773               { fitr }  { fitr }
1774             }
1775             { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1776           \scan_stop:
1777          \tex_pdfdest:D
1778             struct~
1779             \exp_after:wN \__kernel_pdf_object_id_indexed:nn \l_pdf_current_structure_des
1780             name {#1}
1781             \str_case:nnF {#2}
1782               {
1783                 { xyz }   { xyz }
1784                 { fit }   { fit }
1785                 { fitb }  { fitb }
1786                 { fitbh } { fitbh }
1787                 { fitbv } { fitbv }
1788                 { fith }  { fith }
1789                 { fitv }  { fitv }
1790                 { fitr }  { fitr }
1791               }
1792               { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1793             \scan_stop:
1794       }
1795     \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nnnn #1#2#3#4
1796       {
1797        \tex_pdfdest:D
1798         name {#1}
1799         fitr ~
1800         width  \dim_eval:n {#2} ~
1801         height \dim_eval:n {#3} ~
1802         depth  \dim_eval:n {#4} \scan_stop:
1803        \tex_pdfdest:D
1804          struct~
1805          \exp_after:wN \__kernel_pdf_object_id_indexed:nn \l_pdf_current_structure_destinati
1806          name {#1}
1807          fitr ~
```

```
1808          width  \dim_eval:n {#2} ~
1809          height \dim_eval:n {#3} ~
1810          depth  \dim_eval:n {#4} \scan_stop:
1811      }
1812    }
1813 ⟨/pdftex⟩
```

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

```
1814 ⟨*luatex⟩
1815  \int_compare:nNnT {\directlua{tex.print(status.list()["development_id"])} } > {7468}
1816    {
1817      \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nn #1#2
1818        {
1819          \tex_pdfextension:D dest
1820            name {#1}
1821            \str_case:nnF {#2}
1822              {
1823                { xyz }   { xyz }
1824                { fit }   { fit }
1825                { fitb }  { fitb }
1826                { fitbh } { fitbh }
1827                { fitbv } { fitbv }
1828                { fith }  { fith }
1829                { fitv }  { fitv }
1830                { fitr }  { fitr }
1831              }
1832              { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1833            \scan_stop:
1834          \tex_pdfextension:D dest
1835            struct~
1836            \exp_after:wN \__kernel_pdf_object_id_indexed:nn \l_pdf_current_structure_desti
1837            name {#1}
1838            \str_case:nnF {#2}
1839              {
1840                { xyz }   { xyz }
1841                { fit }   { fit }
1842                { fitb }  { fitb }
1843                { fitbh } { fitbh }
1844                { fitbv } { fitbv }
1845                { fith }  { fith }
1846                { fitv }  { fitv }
1847                { fitr }  { fitr }
1848              }
1849              { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1850            \scan_stop:
1851        }
1852      \cs_set_protected:Npn \__pdf_backend_indexed_structure_destination:nnnn #1#2#3#4
1853        {
1854          \tex_pdfextension:D dest
1855            name {#1}
1856            fitr ~
1857            width  \dim_eval:n {#2} ~
1858            height \dim_eval:n {#3} ~
1859            depth  \dim_eval:n {#4} \scan_stop:
1860          \tex_pdfextension:D dest
```

```
1861          struct~
1862          \exp_after:wN \__kernel_pdf_object_id_indexed:nn \l_pdf_current_structure_destinati
1863          name {#1}
1864          fitr ~
1865          width  \dim_eval:n {#2} ~
1866          height \dim_eval:n {#3} ~
1867          depth  \dim_eval:n {#4} \scan_stop:
1868        }
1869      \cs_set_protected:Npn \__pdf_backend_link_begin_structure_goto:nnw #1#2
1870        {
1871          \__pdf_backend_link_begin:nnnw {#1} { goto~struct~name~{#2}~name } {#2}
1872        }
1873    }
1874 ⟨/luatex⟩
```

(*End of definition for* __pdf_backend_indexed_structure_destination:nn *and* __pdf_backend_-
indexed_structure_destination:nnnn.)

1.12 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 dependent part. The main command
is then in l3pdfmeta

```
1875 ⟨*drivers⟩
1876 \cs_new_protected:Npn \__pdf_backend_set_regression_data:
1877    {
1878      \sys_gset_rand_seed:n{1000}
1879      \pdfmanagement_add:nnn{Info}{Creator}{(TeX)}
1880 ⟨/drivers⟩
1881 ⟨*dvips⟩
1882      \AddToHook{begindocument}{\pdfmanagement_add:nnn{Info}{Producer}{(pdfTeX+dvips)}}
1883      \__kernel_backend_literal:e{!~<</DocumentUUID~(DocumentUUID)>>~setpagedevice}
1884      \__kernel_backend_literal:e{!~<</InstanceUUID~(InstanceUUID)>>~setpagedevice}
1885      \str_if_exist:NTF\c_sys_timestamp_str
1886        {
1887          \pdfmanagement_add:nne{Info}{CreationDate}{(\c_sys_timestamp_str)}
1888          \pdfmanagement_add:nne{Info}{ModDate}{(\c_sys_timestamp_str)}
1889        }
1890        {
1891          \pdfmanagement_add:nnn{Info}{CreationDate}{(D:20010101205959-00'00')}
1892          \pdfmanagement_add:nnn{Info}{ModDate}{(D:20010101205959-00'00')}
1893        }
1894 ⟨/dvips⟩
1895 ⟨*dvipdfmx⟩
1896      \pdfmanagement_add:nnn{Info}{Producer}{(dvipdfmx)}
1897      \__kernel_backend_literal:e
1898        {pdf:trailerid [~
1899        <00112233445566778899aabbccddeeff>~
1900        <00112233445566778899aabbccddeeff>~
1901            ]}
1902 ⟨/dvipdfmx⟩
1903 ⟨*xdvipdfmx⟩
1904      \pdfmanagement_add:nnn{Info}{Producer}{(xetex)}
1905      \__kernel_backend_literal:e
```

```
1906        {pdf:trailerid [~
1907          <00112233445566778899aabbccddeeff>~
1908          <00112233445566778899aabbccddeeff>~
1909            ]}
```
⟨/xdvipdfmx⟩
⟨∗pdftex⟩
```
1912      \pdfmanagement_add:nnn{Info}{Producer}{(pdfTeX)}
1913      \tex_pdfsuppressptexinfo:D 7 \scan_stop:
1914      \pdftrailerid{2350CAD05F8A7AF0AA4058486855344F}
```
⟨/pdftex⟩
⟨∗luatex⟩
```
1917      \pdfmanagement_add:nnn{Info}{Producer}{(LuaTeX)}
1918      \tex_pdfvariable:D suppressoptionalinfo 7\relax
1919      \tex_pdfvariable:D trailerid
1920        {[~
1921          <2350CAD05F8A7AF0AA4058486855344F>~
1922          <2350CAD05F8A7AF0AA4058486855344F>~
1923        ]}
```
⟨/luatex⟩
⟨∗drivers⟩
```
1926      \str_if_exist:NF\c_sys_timestamp_str
1927        {
1928        \pdfmanagement_add:nnn{Info}{CreationDate}{(D:20010101205959-00'00')}
1929        \pdfmanagement_add:nnn{Info}{ModDate}{(D:20010101205959-00'00')}
1930        \AddToDocumentProperties[document]{creationdate}{D:20010101205959-00'00'}
1931        \AddToDocumentProperties[document]{moddate}{D:20010101205959-00'00'}
1932        \AddToDocumentProperties[hyperref]{pdfmetadate}{D:20010101205959-00'00'}
1933        \AddToDocumentProperties[hyperref]{pdfdate}{D:20010101205959-00'00'}
1934        }
1935      \AddToDocumentProperties[hyperref]{pdfinstanceid}{uuid:0a57c455-157a-4141-8c19-6237d832f
1936      \AddToDocumentProperties[hyperref]{pdfproducer}{\c_sys_engine_exec_str-NN.NN.NN}
1937      }
```
⟨/drivers⟩

1.13 Uncompressed metadata object stream

The xmp metadata should be written "uncompressed" to pdf. It is not quite clear what exactly that means. Probably it only means that there should be no /Filter key in the stream, but packages like pdfx and hyperref try to suppress object compression too, so we add support for it too. With luatex this is possible by using the uncompressed key word. With pdftex one can change locally the compresslevel. (x)dvipdfmx does it automatically and doesn't need some special command. No solution is known for the dvips route. We need it only once, so we make it special and probably no public interface is needed. It writes an unnamed object so should be referenced directly with \pdf_object_ref_last:

⟨∗luatex⟩
```
1940 \cs_new_protected:Npn \__pdf_backend_metadata_stream:n #1
1941   {
1942     \tex_immediate:D \tex_pdfextension:D obj ~uncompressed~
1943       \__pdf_backend_object_write:nn {stream} {{/Type~/Metadata~/Subtype~/XML}{#1}}
1944   }
```
⟨/luatex⟩
⟨∗pdftex⟩
```
1947 \cs_new_protected:Npn \__pdf_backend_metadata_stream:n #1
```

42

```
1948    {
1949      \group_begin:
1950        \tex_pdfcompresslevel:D  0 \scan_stop:
1951        \tex_immediate:D \tex_pdfobj:D
1952        \__pdf_backend_object_write:nn {stream} {{/Type~/Metadata~/Subtype~/XML}{#1}}
1953      \group_end:
1954    }
1955 ⟨/pdftex⟩
1956 ⟨*xdvipdfmx | dvipdfmx | dvips | dvisvgm⟩
1957 \cs_new_protected:Npn \__pdf_backend_metadata_stream:n #1
1958    {
1959      \pdf_object_unnamed_write:nn {stream}{{/Type~/Metadata~/Subtype~/XML}{#1}}
1960    }
1961 ⟨/xdvipdfmx | dvipdfmx | dvips | dvisvgm⟩
```

1.14 Suppressing deprecated PDF features

/ProcSet, /CharSet and the /Info dictionary are deprecated in PDF 2.0. For the pdf/A-4 standard they must be suppressed. Not every engine is able to do this, but for pdfTeX and luatex we define suitable backend command. /ProcSet is suppressed automatically for pdf version 2.0 starting with in texlive 2023.

__pdf_backend_omit_charset:n   The option to omit /Charset exists already for quite some time for the two engines.

```
1962 ⟨*xdvipdfmx | dvipdfmx | dvips | dvisvgm⟩
1963 \cs_new_protected:Npn \__pdf_backend_omit_charset:n #1 {} %#1 number
1964 ⟨/xdvipdfmx | dvipdfmx | dvips | dvisvgm⟩
1965 ⟨*pdftex⟩
1966 \cs_new_protected:Npn \__pdf_backend_omit_charset:n #1 %#1 number
1967    {
1968      \tex_pdfomitcharset:D = #1 \scan_stop:
1969    }
1970 ⟨/pdftex⟩
1971 ⟨*luatex⟩
1972 \cs_new_protected:Npn \__pdf_backend_omit_charset:n #1 %#1 number
1973    {
1974      \tex_pdfvariable:D omitcharset  = #1 \scan_stop:
1975    }
1976 ⟨/luatex⟩
```

(*End of definition for* __pdf_backend_omit_charset:n.)

__pdf_backend_omit_info:n   The option to suppress the info dictionary will be available in texlive 2023.

```
1977 ⟨*xdvipdfmx | dvipdfmx | dvips | dvisvgm⟩
1978 \cs_new_protected:Npn \__pdf_backend_omit_info:n #1 {} %#1 number
1979 ⟨/xdvipdfmx | dvipdfmx | dvips | dvisvgm⟩
1980 ⟨*pdftex⟩
1981 \bool_lazy_and:nnTF
1982    { \int_compare_p:nNn {\tex_pdftexversion:D } > {139} }
1983    { \int_compare_p:nNn {\tex_pdftexrevision:D } > {24} }
1984    {
1985      \cs_new_protected:Npn \__pdf_backend_omit_info:n #1 %#1 number
1986        {
1987          \pdfomitinfodict = #1 \scan_stop:
1988        }
```

```
1989      }
1990    {
1991      \cs_new_protected:Npn \__pdf_backend_omit_info:n #1 {}%#1 number
1992
1993    }
```
⟨/pdftex⟩
⟨∗luatex⟩
```
1996  \int_compare:nNnTF {\directlua{tex.print(status.list()["development_id"])} } > {7560}
1997    {
1998      \cs_new_protected:Npn \__pdf_backend_omit_info:n #1 %#1 number
1999        {
2000          \tex_pdfvariable:D omitinfodict  = #1 \scan_stop:
2001        }
2002    }
2003    {
2004      \cs_new_protected:Npn \__pdf_backend_omit_info:n #1 {} %#1 number
2005    }
```
⟨/luatex⟩

(*End of definition for* __pdf_backend_omit_info:n*.*)

    With luatex it is for some standards also necessary to suppress the CidSet entry in the fonts (with xetex there seem to be no problem.

__pdf_backend_omit_cidset:n   The option to omit /Charset exists already for quite some time for the two engines.

```
2007  ⟨∗xdvipdfmx | dvipdfmx | dvips | dvisvgm | pdftex⟩
2008  \cs_new_protected:Npn \__pdf_backend_omit_cidset:n #1 {} %#1 number
2009  ⟨/xdvipdfmx | dvipdfmx | dvips | dvisvgm | pdftex⟩
2010  ⟨∗luatex⟩
2011  \cs_new_protected:Npn \__pdf_backend_omit_cidset:n #1 %#1 number
2012    {
2013      \tex_pdfvariable:D omitcidset  = #1 \scan_stop:
2014    }
2015  ⟨/luatex⟩
```

(*End of definition for* __pdf_backend_omit_cidset:n*.*)

1.15   lua code for lualatex

```
2016  ⟨∗lua⟩
2017  ltx= ltx or {}
2018  ltx.__pdf       = ltx.__pdf or {}
2019  ltx.__pdf.Page = ltx.__pdf.Page or {}
2020  ltx.__pdf.Page.dflt = ltx.__pdf.Page.dflt or {}
2021  ltx.__pdf.Page.Resources = ltx.__pdf.Resources or {}
2022  ltx.__pdf.Page.Resources.Properties = ltx.__pdf.Page.Resources.Properties or {}
2023  ltx.__pdf.Page.Resources.List={"ExtGState","ColorSpace","Pattern","Shading"}
2024  ltx.__pdf.object = ltx.__pdf.object or {}
2025
2026  ltx.pdf= ltx.pdf or {} -- for "public" functions
2027
2028  local __pdf = ltx.__pdf
2029  local pdf = pdf
2030
2031  local function __pdf_backend_Page_gput (name,value)
2032    __pdf.Page.dflt[name]=value
```

44

```lua
2033  end
2034
2035  local function __pdf_backend_Page_gremove (name)
2036    __pdf.Page.dflt[name]=nil
2037  end
2038
2039  local function __pdf_backend_Page_gclear ()
2040    __pdf.Page.dflt={}
2041  end
2042
2043  local function __pdf_backend_ThisPage_gput (page,name,value)
2044    __pdf.Page[page] = __pdf.Page[page] or {}
2045    __pdf.Page[page][name]=value
2046  end
2047
2048  local function __pdf_backend_ThisPage_gpush (page)
2049    local token=""
2050    local t = {}
2051    local tkeys= {}
2052    for name,value in pairs(__pdf.Page.dflt) do
2053      t[name]=value
2054    end
2055    if __pdf.Page[page] then
2056      for name,value in pairs(__pdf.Page[page]) do
2057        t[name] = value
2058      end
2059    end
2060    -- sort the table to get reliable test files.
2061    for name,value in pairs(t) do
2062      table.insert(tkeys,name)
2063    end
2064    table.sort(tkeys)
2065    for _,name in ipairs(tkeys) do
2066      token = token .. "/"..name.." "..t[name]
2067    end
2068    return token
2069  end
2070
2071  function ltx.__pdf.backend_ThisPage_gput (page,name,value) -- tex.count["g_shipout_readonly_
2072    __pdf_backend_ThisPage_gput (page,name,value)
2073  end
2074
2075  function ltx.__pdf.backend_ThisPage_gpush (page)
2076    pdf.setpageattributes(__pdf_backend_ThisPage_gpush (page))
2077  end
2078
2079  function ltx.__pdf.backend_Page_gput (name,value)
2080    __pdf_backend_Page_gput (name,value)
2081  end
2082
2083  function ltx.__pdf.backend_Page_gremove (name)
2084    __pdf_backend_Page_gremove (name)
2085  end
2086
```

```lua
function ltx.__pdf.backend_Page_gclear ()
  __pdf_backend_Page_gclear ()
end


local Properties  = ltx.__pdf.Page.Resources.Properties
local ResourceList= ltx.__pdf.Page.Resources.List
local function __pdf_backend_PageResources_gpush (page)
 local token=""
 if Properties[page] then
-- we sort the table, so that the pdf test works
  local t = {}
  for name,value in pairs  (Properties[page]) do
   table.insert (t,name)
  end
  table.sort (t)
  for _,name in ipairs(t) do
   token = token .. "/"..name.." ".. Properties[page][name]
  end
  token = "/Properties <<"..token..">>"
 end
  for i,name in ipairs(ResourceList) do
   if ltx.__pdf.Page.Resources[name] then
   token = token .. "/"..name.." "..ltx.pdf.object_ref("__pdf/Page/Resources/"..name)
   end
  end
 return token
end

-- the function is public, as I probably need it in tagpdf too ...
function ltx.pdf.Page_Resources_Properties_gput (page,name,value) -- tex.count["g_shipout_re
 Properties[page] = Properties[page] or {}
 Properties[page][name]=value
 pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
end

function ltx.pdf.Page_Resources_gpush(page)
 pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
end

function ltx.pdf.object_ref (objname)
 if ltx.__pdf.object[objname] then
  local ref= ltx.__pdf.object[objname]
  return ref
 else
  return "false"
 end
end
⟨/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.

47

51