The **latex-lab-graphic** package
Tagging of included graphics

LATEX Project[*]

v0.80e 2024-09-18

Abstract

The following code implements a first draft for the tagging of graphics included with `\includegraphics`.

1 Introduction

The code here handle the tagging of pictures included with `\includegraphics` and the picture environment. Pictures drawn with `l3draw` or `tikz` or similar packages aren't handled yet.

Tagging of graphics included with `\includegraphics` is at a first glance trivial: They are either only decorations, in which case they should be in a `artifact` MC-chunk or (in pdf 2.0) tagged as an `Artifact` structure, or they are meaningful and then they should be tagged as a `Figure`. Such a graphic is a simple box and no other content can interfere so adding the structure commands shouldn't pose much problems.

But things are actually not so easy.

At first there are two ways to add a graphic to a structure: similar to text as a marked content item (by surrounding it with `\tagmcbegin` and `\tagmcend`) or by referencing the XObject with an OBJR object (similar to a link annotation). Which method is more sensible (and if it actually matters) is unknown but should be tested. Currently the first method is used as the second require changes in the backend files.

At second—and this is actually a *much* larger problem—a `Figure` structure should have an attribute with an `BBox` entry. The value of a `BBox` is an array of four numbers that gives the coordinates of the left, bottom, right, and top edges of the structure element's bounding box. That is the rectangle that completely encloses its *visible* content so not necessarily the TeX bounding box: if `viewport` or `trim` is used and the graphic is not clipped, the visible content can be larger.

Getting the `BBox` is quite straightforward for a graphic that is used once as is. But graphics can be trimmed, scaled, reflected, rotated and reused in various ways. This transformations typically involve a mix of TEX commands like shifting a box or changing the bounding box and backend commands like inserting a pdfliteral with a transformation matrix and and not in all cases getting the `BBox` is possible without rewriting large parts of the graphics/x packages. Problematic are

---

[*]Initial implementation done by Ulrike Fischer

- manipulations through external box commands (`\rotatebox`, `\reflectbox`, `\scalebox`). The current implementation in the graphics/x packages do not pass the transformation matrix in way that allows to track the changes for the `BBox` of an included graphic: sometimes the values are set to late (after the box is already stored), and often the values are not grouped and can leak out from earlier uses of the commands.

- some combination of keys in the optional argument of `\includegraphics`. Examples are `origin` and multiple calls to `scale` and `angle`) as they internally call the box commands. Examples of failing combinations can be found in the test file `graphic-faults`.

- graphics that are stored in a box and reused: to get the `BBox` one has to set a label that stores the position with `\pdfsavepos`, and if a box is reused one gets multiply defined labels. One possible solution here is to make use of the new delayed `\pdfliteral`. It allows to change the label names in the shipout, but this requires careful tracking the box usages and so various kernel changes.

2   Restrictions and Todos

Correct tagging is currently implemented only for simple `\includegraphics` and the keys `viewport`, `trim`, `scale` and `angle` (used at most once).

Not supported

- graphics inside `\rotatebox`, `\reflectbox`, `\scalebox`.

    TODO: A new implementation with `l3graphics` and `l3box` is probably needed here.

- multiple uses of the `scale` and `angle` keys

- multiple use of graphics stored in boxes. For such graphics automated tagging should be probably deactivated when storing the content and tagging should be added around the `\usebox`. (How to proceed when content is saved in boxes needs generally more testing).

3   Additional keys

The code defines additional keys for `\includegraphics`:

`tag` with the values

   artifact When used the graphic will be tagged as artifact. This doesn't require a `BBox` and so works also in some of the not yet supported cases described above.

   false When used tagging will be stopped completely. It is then the responsibility of the surrounding code to add appropriate tagging commands.

   ⟨*name*⟩ Other values will be used as tag names in the structure. If the tag is not known as a structure tag you will get an warning from tagpdf. The default name is currently `Figure`

actualtext This allows to add an /ActualText to the structure. This is useful for small graphics that represent single chars or a short word like a logo. If `actualtext` is used, the graphics is not enclosed in `Figure` structure but in a `Span` structure and no /BBox attribute is added. This in accordance with (the draft of) PDF/UA-2 but violates perhaps PDF/UA-1.

correct-BBox If the calculated /BBox values are wrong they can be correct with this key. It expects four dimensions that are added to the /BBox values.

debug The value BBox will show the calculated /BBox as a half transparent red rectangle.

The code also redefines the `alt` key to actually add its values as an alternative text. If no `alt` value is given, a warning is issued and the file name of the graphic is used.

```
1 ⟨@@=tag⟩
2 ⟨*package⟩
```

4 Implementation

```
3 \ProvidesExplPackage {latex-lab-testphase-graphic} {\ltlabgraphicdate} {\ltlabgraphicversion}
4    {Code related to the tagging of graphics}
```

We load l3opacity for the debug code

```
5 \RequirePackage{l3opacity}
```

Needed during switch to e-type:

```
6 \cs_generate_variant:Nn \__tag_prop_gput:Nnn {cne}
```

__tag_graphic_savepos:n   this is the command which stores the position. Similar to zref-savepos it uses two savepos commands for the case that bidi changes the processing order.

```
7  \cs_new_protected:Npn\__tag_graphic_savepos:n #1
8  {
9    \tex_savepos:D
10   \property_record:nn{#1}{xpos,ypos,abspage}
11   \tex_savepos:D
12  }
13 \cs_generate_variant:Nn \__tag_graphic_savepos:n {e}
```

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

4.1 Variables

\l__tag_graphic_debug_bool   A boolean for debug code

```
14 \bool_new:N \l__tag_graphic_debug_bool
15 \keys_define:nn { document / metadata }
16   {
17     debug / BBox .code:n = { \bool_set_true:N \l__tag_graphic_debug_bool }
18   }
```

(*End of definition for* \l__tag_graphic_debug_bool.)

\g__tag_graphic_int   This is used to get unique labels in the savepos code.

```
19 \int_new:N\g__tag_graphic_int
```

(*End of definition for* \g__tag_graphic_int.)

<div style="float:left">

`\g__tag_graphic_lx_tl`
`\g__tag_graphic_ly_tl`
`\g__tag_graphic_ux_tl`
`\g__tag_graphic_uy_tl`
`bboxcorr_seq␣␣\l_tag_graphic_bboxcorr_bool`

</div>

This commands will hold the calculated BBox values. Local variables would probably work too, but global variables can be easier retrieved in tests and debugging code ...

```
20 \tl_new:N \g__tag_graphic_lx_tl
21 \tl_new:N \g__tag_graphic_ly_tl
22 \tl_new:N \g__tag_graphic_ux_tl
23 \tl_new:N \g__tag_graphic_uy_tl
24 \seq_new:N\l__tag_graphic_bboxcorr_seq
25 \bool_new:N\l__tag_graphic_bboxcorr_bool
```

(*End of definition for* `\g__tag_graphic_lx_tl` *and others.*)

`\l__tag_graphic_currentlabel_tl`   This holds the label name of the savepos.

```
26 \tl_new:N \l__tag_graphic_currentlabel_tl
```

(*End of definition for* `\l__tag_graphic_currentlabel_tl`.)

<div style="float:left">

`\l__tag_graphic_alt_tl`
`\l__tag_graphic_alt_dft_tl`
`\l__tag_graphic_actual_tl`
`\l__tag_graphic_struct_tl`
`\l__tag_graphic_artifact_bool`
`\l__tag_graphic_BBox_bool`

</div>

Variables for the alt text, the actualtext and the structure tag.

```
27 \tl_new:N \l__tag_graphic_alt_tl
28 \tl_new:N \l__tag_graphic_alt_dflt_tl
29 \tl_set:Nn \l__tag_graphic_alt_dflt_tl {\Gin@base\Gin@ext}
30 \tl_new:N \l__tag_graphic_actual_tl
31 \tl_new:N \l__tag_graphic_struct_tl
32 \tl_set:Nn\l__tag_graphic_struct_tl {Figure}
33 \bool_new:N\l__tag_graphic_artifact_bool
34 \bool_new:N\l__tag_graphic_BBox_bool
35 \bool_set_true:N\l__tag_graphic_BBox_bool
```

(*End of definition for* `\l__tag_graphic_alt_tl` *and others.*)

<div style="float:left">

`\l__tag_graphic_sin_fp`
`\l__tag_graphic_cos_fp`
`\l__tag_graphic_scale_fp`
`\l__tag_graphic_lxly_fp`
`\l__tag_graphic_lxuy_fp`
`\l__tag_graphic_uxly_fp`
`\l__tag_graphic_uxuy_fp`
`\l__tag_graphic_ux_fp`
`\l__tag_graphic_ly_fp`
`\l__tag_graphic_lx_fp`
`\l__tag_graphic_uy_fp`
`\l__tag_graphic_trim_ux_fp`
`\l__tag_graphic_trim_ly_fp`
`\l__tag_graphic_trim_lx_fp`
`\l__tag_graphic_trim_uy_fp`

</div>

A bunch of fp-variables (we don't use tl-vars, to avoid to have to take care about minus signs everywhere)

```
36 \fp_new:N\l__tag_graphic_sin_fp
37 \fp_new:N\l__tag_graphic_cos_fp
38 \fp_new:N\l__tag_graphic_lxly_fp
39 \fp_new:N\l__tag_graphic_lxuy_fp
40 \fp_new:N\l__tag_graphic_uxly_fp
41 \fp_new:N\l__tag_graphic_uxuy_fp
42 \fp_new:N\l__tag_graphic_ux_fp
43 \fp_new:N\l__tag_graphic_ly_fp
44 \fp_new:N\l__tag_graphic_lx_fp
45 \fp_new:N\l__tag_graphic_uy_fp
```

this holds the scale value. Either `\Gin@scalex` or (if that is !) `\Gin@scaley`

```
46 \fp_new:N\l__tag_graphic_scale_fp
```

the follow variables hold the four trim values (or the equivalent calculated values if viewport is used.

```
47 \fp_new:N\l__tag_graphic_trim_ux_fp
48 \fp_new:N\l__tag_graphic_trim_ly_fp
49 \fp_new:N\l__tag_graphic_trim_lx_fp
50 \fp_new:N\l__tag_graphic_trim_uy_fp
```

(*End of definition for* `\l__tag_graphic_sin_fp` *and others.*)

4.2 Tagging commands

`\Gin@tag@struct@begin` The command to start the tagging.

```
51 \msg_new:nnn {tag}{alt-text-missing}
52   {
53     Alternative~text~for~graphic~is~missing.\\
54     Using~the~file~name~'#1'~instead.
55   }
56 \cs_new_protected:Npn\Gin@tag@struct@begin
57   {
58     \tag_if_active:T
59       {
60         \tag_mc_end_push:
```

we don't open a structure for artifacts to make it easier to use graphics in saveboxes.

```
61         \bool_if:NTF\l__tag_graphic_artifact_bool
62          {
63            \tag_mc_begin:n{artifact}
64          }
65          {
66            \tl_if_empty:NTF\l__tag_graphic_actual_tl
67              {
68                \tl_if_empty:NT\l__tag_graphic_alt_tl
69                  {
70                    \msg_warning:nne{tag}{alt-text-missing}{\l__tag_graphic_alt_dflt_tl}
71                    \tl_set:Ne\l__tag_graphic_alt_tl {\l__tag_graphic_alt_dflt_tl}
72                  }
73                \tag_struct_begin:n
74                  {
75                    tag=\l__tag_graphic_struct_tl,
76                    alt=\l__tag_graphic_alt_tl,
77                  }
78              }
79              {
80                \tag_struct_begin:n
81                  {
82                    tag=Span,
83                    actualtext=\l__tag_graphic_actual_tl,
84                  }
85                \bool_set_false:N\l__tag_graphic_BBox_bool
86              }
87            \tag_mc_begin:n{}
88          }
89       }
90   }
```

(End of definition for `\Gin@tag@struct@begin`*. This function is documented on page* **??***.)*

`\Gin@tag@struct@end`

```
91 \cs_new_protected:Npn\Gin@tag@struct@end
92   {
93     \tag_if_active:T
94       {
95         \tag_mc_end:
96         \bool_if:NF\l__tag_graphic_artifact_bool
```

5

```
97        {
98          \tag_struct_end:
99        }
100      \tag_mc_begin_pop:n{}
101    }
102  }
```

(*End of definition for* \Gin@tag@struct@end. *This function is documented on page* **??**.)

4.3 Patching graphics commands

All changes are currently done in \Gin@setfile.

```
103  \AddToHook{package/graphics/after}
104  {
105    \def\Gin@setfile#1#2#3{%
106    \ifx\\#2\\\Gread@false\fi
107    \ifGin@bbox\else
108      \ifGread@
109        \csname Gread@%
110          \expandafter\ifx\csname Gread@#1\endcsname\relax
111            eps%
112          \else
113            #1%
114          \fi
115        \endcsname{\Gin@base#2}%
116      \else
117        \Gin@nosize{#3}%
118      \fi
119    \fi
120    \Gin@viewport@code
121    \Gin@nat@height\Gin@ury bp%
122    \advance\Gin@nat@height-\Gin@lly bp%
123    \Gin@nat@width\Gin@urx bp%
124    \advance\Gin@nat@width-\Gin@llx bp%
125    \Gin@req@sizes
126    \expandafter\ifx\csname Ginclude@#1\endcsname\relax
127      \Gin@drafttrue
128      \expandafter\ifx\csname Gread@#1\endcsname\relax
129        \@latex@error{Can not include graphics of type: #1}\@ehc
130        \global\expandafter\let\csname Gread@#1\endcsname\@empty
131      \fi
132    \fi
133    \leavevmode
```

Here the tagging begins. We want to catch also the draft box, and for luatex tagging must be started before the \setbox.

```
134      \Gin@tag@struct@begin %NEW
135      \ifGin@draft
136        \hb@xt@\Gin@req@width{%
137          \vrule\hss
138          \vbox to \Gin@req@height{%
139            \hrule \@width \Gin@req@width
140            \vss
141            \edef\@tempa{#3}%
```

```
142        \rlap{ \ttfamily\expandafter\strip@prefix\meaning\@tempa}%
143          \vss
144          \hrule}%
145        \hss\vrule}%
146    \else
147      \@addtofilelist{#3}%
148      \ProvidesFile{#3}[Graphic file (type #1)]%
149      \setbox\z@\hbox{\csname Ginclude@#1\endcsname{#3}}%
150      \dp\z@\z@
151      \ht\z@\Gin@req@height
152      \wd\z@\Gin@req@width
```

This the main command to calculate the BBox values.

```
153      \Gin@tag@bbox@attribute %new
154      \box\z@
```

and here the tagging stops.

```
155    \fi
156    \Gin@tag@struct@end %new
157    }
158  }
```

4.4   Additional keys for the graphics command

TODO: this is a bit temporary and will perhaps need more refinement. we also ensure
that graphicx is loaded for the keyval support.

```
159  \AddToHook{package/graphicx/after}[latex-lab]
160    {
161      \define@key{Gin}{alt}        {\tl_set:Ne\l__tag_graphic_alt_tl{\text_purify:n{#1}}}
162      \define@key{Gin}{artifact}[]
163        {
164          \bool_set_true:N \l__tag_graphic_artifact_bool
165          \bool_set_false:N \l__tag_graphic_BBox_bool
166        }
167      \define@key{Gin}{actualtext}
168       {
169          \tl_set:Ne\l__tag_graphic_actual_tl{\text_purify:n{#1}}
170          \bool_set_false:N \l__tag_graphic_BBox_bool
171       }
172      \define@key{Gin}{correct-BBox}
173        {
174          \bool_set_true:N \l__tag_graphic_bboxcorr_bool
175          \seq_set_split:Nnn\l__tag_graphic_bboxcorr_seq{~}{#1~0pt~0pt~0pt~0pt}
176        }
177      \define@key{Gin}{tag}
178        {
179          \str_case:nnF {#1}
180            {
181              {artifact}
182              {
183                \bool_set_true:N \l__tag_graphic_artifact_bool
184                \bool_set_false:N \l__tag_graphic_BBox_bool
185              }
186              {false}{\tag_suspend:n{Gin}}
```

7

```
187            }
188            {\tl_set:Nn\l__tag_graphic_struct_tl{#1}}
189        }
190     }
191  \AddToHook{package/graphics/after}[latex-lab]
192    {\RequirePackage{graphicx}}
```

For picture and other environments we need a similar set of keys. TODO: redefine \includegraphics to make use of these here??

```
193  \keys_define:nn{tag/picture}
194     {
195       ,alt .code:n =
196          {\tl_set:Ne\l__tag_graphic_alt_tl{\text_purify:n{#1}}}
197       ,artifact .code:n =
198          {
199            \bool_set_true:N \l__tag_graphic_artifact_bool
200            \bool_set_false:N \l__tag_graphic_BBox_bool
201          }
202       ,actualtext .code:n =
203          {
204            \tl_set:Ne\l__tag_graphic_actual_tl{\text_purify:n{#1}}
205            \bool_set_false:N \l__tag_graphic_BBox_bool
206          }
207       ,correct-BBox .code:n =
208         {
209           \bool_set_true:N \l__tag_graphic_bboxcorr_bool
210           \seq_set_split:Nnn\l__tag_graphic_bboxcorr_seq{~}{#1~0pt~0pt~0pt~0pt}
211         }
212       ,tag .code:n =
213          {
214            \str_case:nnF {#1}
215              {
216                 {artifact}
217                 {
218                    \bool_set_true:N \l__tag_graphic_artifact_bool
219                    \bool_set_false:N \l__tag_graphic_BBox_bool
220                 }
221              {false}{\tag_suspend:n{picture}}
222              }
223            {\tl_set:Nn\l__tag_graphic_struct_tl{#1}}
224        }
225     }
```

4.5   Calculating the BBox

__tag_graphic_get_trim: Graphics can be trimmed with the trim and the viewport key. If the graphic is not clipped the values must be taken into account when rotating. If viewport is used we have to calculate the trim.

```
226  \cs_new_protected:Npn \__tag_graphic_get_trim:
227  {
228     \legacy_if:nTF {Gin@clip}
```

Setting to 0 is not strictly needed but looks cleaner.

```
229       {
```

```
230        \fp_zero:N\l__tag_graphic_trim_lx_fp
231        \fp_zero:N\l__tag_graphic_trim_ly_fp
232        \fp_zero:N\l__tag_graphic_trim_ux_fp
233        \fp_zero:N\l__tag_graphic_trim_uy_fp
234      }
235      {
236        \fp_set:Nn \l__tag_graphic_trim_lx_fp {\l__tag_graphic_scale_fp*\Gin@vllx}
237        \fp_set:Nn \l__tag_graphic_trim_ly_fp {\l__tag_graphic_scale_fp*\Gin@vlly}
238        \fp_set:Nn \l__tag_graphic_trim_ux_fp {\l__tag_graphic_scale_fp*\Gin@vurx}
239        \fp_set:Nn \l__tag_graphic_trim_uy_fp {\l__tag_graphic_scale_fp*\Gin@vury}
240        \cs_if_exist:NT \Gin@ollx
241          {
242           \fp_set:Nn \l__tag_graphic_trim_ux_fp {\l__tag_graphic_scale_fp* (\Gin@ourx-(\Gin@ur
243           \fp_set:Nn \l__tag_graphic_trim_uy_fp {\l__tag_graphic_scale_fp* (\Gin@oury-(\Gin@ur
244          }
245      }
246  }
```

(End of definition for __tag_graphic_get_trim:.)

__tag_graphic_get_scale:

```
247  \cs_new_protected:Npn \__tag_graphic_get_scale:
248  {
249    \fp_set:Nn \l__tag_graphic_scale_fp
250      {
251        \str_if_eq:eeTF {\Gin@scalex} { ! }
252        { \Gin@scaley }
253        { \Gin@scalex }
254      }
255  }
```

(End of definition for __tag_graphic_get_scale:.)

__tag_graphic_applyangle:nnnn   This takes the current BBox and rotates it according to the use angle. This is the most
laborious code, as we have to take also the trim values into account. We have to compare
the values after the rotation to find the right corners for the BBox. Not sure, if this is
the most effective code, the l3draw package has similar code to calculate a rotation, this
can perhaps be reused …

```
256  \cs_new_protected:Npn \__tag_graphic_applyangle:nnnn #1#2#3#4 %lx,ly,ux,uy
257  {
258    \bool_lazy_and:nnT
259      {\cs_if_exist_p:N \Grot@angle }
260      {! \int_compare_p:nNn { \Grot@angle }={0}}
261      {
262        \fp_set:Nn \l__tag_graphic_sin_fp  { sind(\Grot@angle) }
263        \fp_set:Nn \l__tag_graphic_cos_fp  { cosd(\Grot@angle) }
264        \fp_set:Nn \l__tag_graphic_lx_fp {#1}
265        \fp_set:Nn \l__tag_graphic_ly_fp {#2}
266        \fp_set:Nn \l__tag_graphic_ux_fp {#3}
267        \fp_set:Nn \l__tag_graphic_uy_fp {#4}
```

get the x coordinates (cos,-sin)

```
268        \fp_set:Nn\l__tag_graphic_lxly_fp
269          {
```

```
270        -\l__tag_graphic_trim_lx_fp *  \l__tag_graphic_cos_fp
271        +\l__tag_graphic_trim_ly_fp *  \l__tag_graphic_sin_fp
272      }
273    \fp_set:Nn\l__tag_graphic_lxuy_fp
274      {
275        (-\l__tag_graphic_trim_lx_fp) * \l__tag_graphic_cos_fp
276        +
277         (\l__tag_graphic_uy_fp-\l__tag_graphic_ly_fp-\l__tag_graphic_trim_ly_fp)
278         * (-\l__tag_graphic_sin_fp)
279      }
280    \fp_set:Nn\l__tag_graphic_uxly_fp
281      {
282        (\l__tag_graphic_ux_fp-\l__tag_graphic_lx_fp-\l__tag_graphic_trim_lx_fp)
283         * \l__tag_graphic_cos_fp
284        +
285        (\l__tag_graphic_trim_ly_fp) * (\l__tag_graphic_sin_fp)
286      }
287    \fp_set:Nn\l__tag_graphic_uxuy_fp
288      {
289        (\l__tag_graphic_ux_fp-\l__tag_graphic_lx_fp-\l__tag_graphic_trim_lx_fp)
290         *  \l__tag_graphic_cos_fp
291       +
292        (\l__tag_graphic_uy_fp-\l__tag_graphic_ly_fp-\l__tag_graphic_trim_ly_fp)
293         * (-\l__tag_graphic_sin_fp)
294      }
295    \tl_gset:Ne\g__tag_graphic_lx_tl
296      {
297       \fp_eval:n
298         {
299           min
300            (
301              \l__tag_graphic_lxly_fp,
302              \l__tag_graphic_lxuy_fp,
303              \l__tag_graphic_uxly_fp,
304              \l__tag_graphic_uxuy_fp,
305            )
306            +\l__tag_graphic_lx_fp
307            +\l__tag_graphic_trim_lx_fp
308         }
309      }
310    \tl_gset:Ne\g__tag_graphic_ux_tl
311      {
312       \fp_eval:n
313         {
314           max
315            (
316              \l__tag_graphic_lxly_fp,
317              \l__tag_graphic_lxuy_fp,
318              \l__tag_graphic_uxly_fp,
319              \l__tag_graphic_uxuy_fp
320            )
321            +\l__tag_graphic_lx_fp
322            +\l__tag_graphic_trim_lx_fp
323          }
```

```
324              }
```

get the y coordinates (sin,cos)

```
325          \fp_set:Nn\l__tag_graphic_lxly_fp
326           {
327             -\l__tag_graphic_trim_lx_fp *  \l__tag_graphic_sin_fp
328             -\l__tag_graphic_trim_ly_fp *  \l__tag_graphic_cos_fp
329           }
330          \fp_set:Nn\l__tag_graphic_lxuy_fp
331           {
332             - \l__tag_graphic_trim_lx_fp *  \l__tag_graphic_sin_fp
333             +
334             (\l__tag_graphic_uy_fp-\l__tag_graphic_ly_fp-\l__tag_graphic_trim_ly_fp)
335              * \l__tag_graphic_cos_fp
336           }
337          \fp_set:Nn\l__tag_graphic_uxly_fp
338           {
339             (\l__tag_graphic_ux_fp-\l__tag_graphic_lx_fp-\l__tag_graphic_trim_lx_fp)
340              * \l__tag_graphic_sin_fp
341             - \l__tag_graphic_trim_ly_fp * \l__tag_graphic_cos_fp
342           }
343          \fp_set:Nn\l__tag_graphic_uxuy_fp
344           {
345             (\l__tag_graphic_ux_fp-\l__tag_graphic_lx_fp-\l__tag_graphic_trim_lx_fp)
346              * \l__tag_graphic_sin_fp
347            +
348             (\l__tag_graphic_uy_fp-\l__tag_graphic_ly_fp-\l__tag_graphic_trim_ly_fp)
349              * \l__tag_graphic_cos_fp
350           }
351          \tl_gset:Ne\g__tag_graphic_ly_tl
352            {
353            \fp_eval:n
354              {
355                min
356                 (
357                   \l__tag_graphic_lxly_fp,
358                   \l__tag_graphic_lxuy_fp,
359                   \l__tag_graphic_uxly_fp,
360                   \l__tag_graphic_uxuy_fp
361                 )
362                + \l__tag_graphic_ly_fp + \l__tag_graphic_trim_ly_fp
363              }
364            }
365          \tl_gset:Ne\g__tag_graphic_uy_tl
366            {
367            \fp_eval:n
368              {
369                max
370                 (
371                   \l__tag_graphic_lxly_fp,
372                   \l__tag_graphic_lxuy_fp,
373                   \l__tag_graphic_uxly_fp,
374                   \l__tag_graphic_uxuy_fp,
375                 )
376                + \l__tag_graphic_ly_fp + \l__tag_graphic_trim_ly_fp
```

```
377                    }
378                }
379            }
380    }
381 \cs_generate_variant:Nn\__tag_graphic_applyangle:nnnn {VVVV}
```

(*End of definition for* `\__tag_graphic_applyangle:nnnn`.)

`\__tag_graphic_applycorr:NNNN`   This command is used to add at the end the correction values. Quite dump ...

```
382 \cs_new_protected:Npn \__tag_graphic_applycorr:NNNN #1 #2 #3 #4
383    {
384      \bool_if:NT\l__tag_graphic_bboxcorr_bool
385        {
386          \tl_set:Ne #1
387            {
388              \fp_eval:n
389                {
390                  #1
391                  +
392                  \dim_to_decimal_in_bp:n {\seq_item:Nn \l__tag_graphic_bboxcorr_seq {1} }
393                }
394            }
395          \tl_set:Ne #2
396            {
397              \fp_eval:n
398                {
399                  #2
400                  +
401                  \dim_to_decimal_in_bp:n {\seq_item:Nn \l__tag_graphic_bboxcorr_seq {2} }
402                }
403            }
404          \tl_set:Ne #3
405            {
406              \fp_eval:n
407                {
408                  #3
409                  +
410                  \dim_to_decimal_in_bp:n {\seq_item:Nn \l__tag_graphic_bboxcorr_seq {3} }
411                }
412            }
413          \tl_set:Ne #4
414            {
415              \fp_eval:n
416                {
417                  #4
418                  +
419                  \dim_to_decimal_in_bp:n {\seq_item:Nn \l__tag_graphic_bboxcorr_seq {4} }
420                }
421            }
422        }
423    }
```

(*End of definition for* `\__tag_graphic_applycorr:NNNN`.)

\Gin@tag@bbox@attribute   This is the main command to calculate and set the Bbox attribute

```
424 \cs_new_protected:Npn \Gin@tag@bbox@attribute
425    {
```

the attribute is only needed if tagging is active and there is not artifact.

```
426    \bool_lazy_all:nT
427      {
428        {\tag_if_active_p:}
429        {!\l__tag_graphic_artifact_bool}
430        {\l__tag_graphic_BBox_bool}
431      }
432      {
433        \__tag_graphic_get_scale:
434        \__tag_graphic_get_trim:
435        \int_gincr:N\g__tag_graphic_int
436        \tl_set:Ne\l__tag_graphic_currentlabel_tl {__tag_graphic_\int_use:N \g__tag_graphic_int
437        \__tag_graphic_savepos:e { \l__tag_graphic_currentlabel_tl }
438        \tl_gset:Ne\g__tag_graphic_lx_tl
439          {
440            \dim_to_decimal_in_bp:n
441              { \property_ref:een {\l__tag_graphic_currentlabel_tl}{xpos}{0}sp }
442          }
443        \tl_gset:Ne\g__tag_graphic_ly_tl
444          {
445            \dim_to_decimal_in_bp:n
446              { \property_ref:een {\l__tag_graphic_currentlabel_tl}{ypos}{0}sp }
447          }
448        \tl_gset:Ne\g__tag_graphic_ux_tl
449            {
450              \fp_eval:n
451                {
452                  \g__tag_graphic_lx_tl
453                  +
454                  \dim_to_decimal_in_bp:n { \Gin@req@width }
455                }
456            }
457         \tl_gset:Ne\g__tag_graphic_uy_tl
458            {
459              \fp_eval:n
460                {
461                  \g__tag_graphic_ly_tl
462                  +
463                   \dim_to_decimal_in_bp:n  { \Gin@req@height }
464                }
465            }
```

If the graphics is not clipped we must add the trim values.

```
466        \legacy_if:nF {Gin@clip}
467          {
468            \tl_gset:Ne\g__tag_graphic_ux_tl
469              {
470                \fp_eval:n
471                  {
472                    \g__tag_graphic_ux_tl
473                    +
```

13

```
474                         \l__tag_graphic_trim_ux_fp
475                       }
476                   }
477               \tl_gset:Ne\g__tag_graphic_lx_tl
478                 {
479                   \fp_eval:n
480                     {
481                       \g__tag_graphic_lx_tl
482                       -
483                       \l__tag_graphic_trim_lx_fp
484                     }
485                 }
486               \tl_gset:Ne\g__tag_graphic_uy_tl
487                 {
488                   \fp_eval:n
489                     {
490                       \g__tag_graphic_uy_tl
491                       +
492                       \l__tag_graphic_trim_uy_fp
493                     }
494                 }
495               \tl_gset:Ne\g__tag_graphic_ly_tl
496                 {
497                   \fp_eval:n
498                     {
499                       \g__tag_graphic_ly_tl
500                       -
501                       \l__tag_graphic_trim_ly_fp
502                     }
503                 }
504             }
```

If there is an angle we now rotate the values.

```
505           \__tag_graphic_applyangle:VVVV
506             \g__tag_graphic_lx_tl
507             \g__tag_graphic_ly_tl
508             \g__tag_graphic_ux_tl
509             \g__tag_graphic_uy_tl
```

At last we have to add the correction values

```
510           \__tag_graphic_applycorr:NNNN
511             \g__tag_graphic_lx_tl
512             \g__tag_graphic_ly_tl
513             \g__tag_graphic_ux_tl
514             \g__tag_graphic_uy_tl
515       \bool_if:NT\l__tag_graphic_debug_bool
516         {
517           \__tag_graphic_show_bbox:VVVVne
518             \g__tag_graphic_lx_tl
519             \g__tag_graphic_ly_tl
520             \g__tag_graphic_ux_tl
521             \g__tag_graphic_uy_tl
522             {red}
523             {\int_use:N\g__tag_graphic_int}
524         }
```

14

Now we add the attribute. We do it manually as it had to be delayed until now. The structure and the mc must be open earlier, before the \setbox (at least for luatex it has to). TODO: think about interface if more attributes are needed.

```
525       \__tag_prop_gput:cne
526         { g__tag_struct_\int_eval:n {\c@g__tag_struct_abs_int}_prop }
527         { A }
528         {
529           <<
530             /O /Layout /BBox~
531             [
532              \g__tag_graphic_lx_tl\c_space_tl
533              \g__tag_graphic_ly_tl\c_space_tl
534              \g__tag_graphic_ux_tl\c_space_tl
535              \g__tag_graphic_uy_tl
536             ]
537           >>
538         }
539     }
540   }
```

(*End of definition for* \Gin@tag@bbox@attribute. *This function is documented on page* **??**.)

4.6   Support for the picture environment

\picture@tag@bbox@attribute   Picture needs a similar command to calculate the bbox. But here we stay simple and use simply the size of the picbox.

```
541 \newcommand\picture@tag@bbox@attribute
542  {
543    \bool_lazy_all:nT
544      {
545        {\tag_if_active_p:}
546        {!\l__tag_graphic_artifact_bool}
547        {\l__tag_graphic_BBox_bool}
548      }
549      {
550        \int_gincr:N\g__tag_graphic_int
551        \tl_set:Ne\l__tag_graphic_currentlabel_tl {__tag_graphic_\int_use:N \g__tag_graphic_int
552        \__tag_graphic_savepos:e { \l__tag_graphic_currentlabel_tl }
553        \tl_gset:Ne \g__tag_graphic_lx_tl
554          {
555            \dim_to_decimal_in_bp:n
556            { \property_ref:een {\l__tag_graphic_currentlabel_tl}{xpos}{0}sp  }
557          }
558        \tl_gset:Ne \g__tag_graphic_ly_tl
559          {
560            \dim_to_decimal_in_bp:n
561            { \property_ref:een {\l__tag_graphic_currentlabel_tl}{ypos}{0}sp - \dp\@picbox }
562          }
563        \tl_gset:Ne \g__tag_graphic_ux_tl
564          {
565            \dim_to_decimal_in_bp:n
566              {
567                \g__tag_graphic_lx_tl bp + \wd\@picbox
```

15

```
568                 }
569               }
570         \tl_gset:Ne \g__tag_graphic_uy_tl
571           {
572             \dim_to_decimal_in_bp:n
573               {
574                 \g__tag_graphic_ly_tl bp + \ht\@picbox + \dp\@picbox
575               }
576           }
577         \__tag_graphic_applycorr:NNNN
578             \g__tag_graphic_lx_tl
579             \g__tag_graphic_ly_tl
580             \g__tag_graphic_ux_tl
581             \g__tag_graphic_uy_tl
582         \bool_if:NT\l__tag_graphic_debug_bool
583           {
584             \__tag_graphic_show_bbox:VVVVne
585             \g__tag_graphic_lx_tl
586             \g__tag_graphic_ly_tl
587             \g__tag_graphic_ux_tl
588             \g__tag_graphic_uy_tl
589             {red}
590             {\int_use:N\g__tag_graphic_int}
591           }
592         \__tag_prop_gput:cne
593             { g__tag_struct_\int_eval:n {\c@g__tag_struct_abs_int}_prop }
594             { A }
595             {
596               <<
597                 /O /Layout /BBox~
598                 [
599                 \g__tag_graphic_lx_tl\c_space_tl
600                 \g__tag_graphic_ly_tl\c_space_tl
601                 \g__tag_graphic_ux_tl\c_space_tl
602                 \g__tag_graphic_uy_tl
603                 ]
604               >>
605             }
606       }
607   }
608
```

(*End of definition for* `\picture@tag@bbox@attribute`*. This function is documented on page* **??**.)

We redefine `\picture` to accept an optional argument and change the default alt text. We also ensure that we are in hmode, so that stopping tagging doesn't confuse the paratags.

```
609 \RenewDocumentCommand\picture{O{}m}
610 {
611   \leavevmode
612   \keys_set:nn{tag/picture}{#1} %
613   \tl_set:Nn\l__tag_graphic_alt_dflt_tl {picture~environment}
614   \pictur@#2
615 }
```

inside the picture box we stop tagging.

```
616 \def\@picture(#1,#2)(#3,#4){%
617   \@defaultunitsset\@picht{#2}\unitlength
618   \@defaultunitsset\@tempdimc{#1}\unitlength
619   \Gin@tag@struct@begin
620   \setbox\@picbox\hb@xt@\@tempdimc\bgroup
621     \tag_suspend:n{\@picture} %do not tag inside the picture box
622     \@defaultunitsset\@tempdimc{#3}\unitlength
623     \hskip -\@tempdimc
624     \@defaultunitsset\@tempdimc{#4}\unitlength
625     \lower\@tempdimc\hbox\bgroup
626       \ignorespaces}
627 \def\endpicture{%
628   \egroup\hss\egroup
629   \ht\@picbox\@picht\dp\@picbox\z@
630   \picture@tag@bbox@attribute
631   \mbox{\box\@picbox}
632   \Gin@tag@struct@end}
```

4.7 Debugging code

__tag_graphic_show_bbox:nnnnnn

```
633 \cs_new_protected:Npn \__tag_graphic_show_bbox:nnnnnn #1#2#3#4#5#6%#5 color, #6 graphic
634   {
635     \iow_log:n {tag/graphic~debug:~BBox~of~graphics~#6~is~#1~#2~#3~#4}
636     \hook_gput_code:nnn
637      {shipout/foreground}
638      {tag/graphic}
639      {
640       \int_compare:nNnT
641        {\g_shipout_readonly_int}
642        =
643        {\property_ref:een{__tag_graphic_#6}{abspage}{0}}
644        {
645         \put
646          (#1 bp,\dim_eval:n{-\paperheight + \dim_eval:n{#2 bp}})
647          {
648           \opacity_select:n{0.5}\color_select:n{#5}
649           \rule
650            {\dim_eval:n {#3 bp-\dim_eval:n{#1 bp}}}
651            {\dim_eval:n {#4 bp-\dim_eval:n{#2 bp}}}
652          }
653        }
654      }
655   }
656 \cs_generate_variant:Nn  \__tag_graphic_show_bbox:nnnnnn {VVVVne}
```

(*End of definition for __tag_graphic_show_bbox:nnnnnn.*)

```
657 ⟨/package⟩
658 ⟨*latex-lab⟩
659 \ProvidesFile{graphic-latex-lab-testphase.ltx}
660         [\ltlabgraphicdate\space v\ltlabgraphicversion\space latex-lab wrapper graphic]
```

```
661 \RequirePackage{latex-lab-testphase-graphic}
662 ⟨/latex-lab⟩
```