

# The latex-lab-graphic package

## Tagging of included graphics

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

v0.80b 2023-10-13

### 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 `\tagmcbend`) 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} {\ltxlabgraphicdate} {\ltxlabgraphicversion}
4 {Code related to the tagging of graphics}
```

We load `l3opacity` for the debug code

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

`\__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.

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

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

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

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

```
18 \int_new:N \g__tag_graphic_int
```

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

`\g__tag_graphic_lx_tl` 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 ...

`\g__tag_graphic_ly_tl`

`\g__tag_graphic_ux_tl` 19 `\tl_new:N \g__tag_graphic_lx_tl`

`\g__tag_graphic_uy_tl` 20 `\tl_new:N \g__tag_graphic_ly_tl`

`\l__tag_graphic_bboxcorr_bool` 21 `\tl_new:N \g__tag_graphic_ux_tl`

22 `\tl_new:N \g__tag_graphic_uy_tl`

23 `\seq_new:N \l__tag_graphic_bboxcorr_seq`

24 `\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.

25 `\tl_new:N \l__tag_graphic_currentlabel_tl`

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

`\l__tag_graphic_alt_tl` Variables for the alt text, the actualtext and the structure tag.

`\l__tag_graphic_alt_dflt_tl` 26 `\tl_new:N \l__tag_graphic_alt_tl`

`\l__tag_graphic_actual_tl` 27 `\tl_new:N \l__tag_graphic_alt_dflt_tl`

`\l__tag_graphic_struct_tl` 28 `\tl_set:Nn \l__tag_graphic_alt_dflt_tl {\Gin@base\Gin@ext}`

`\l__tag_graphic_artifact_bool` 29 `\tl_new:N \l__tag_graphic_actual_tl`

`\l__tag_graphic_BBox_bool` 30 `\tl_new:N \l__tag_graphic_struct_tl`

31 `\tl_set:Nn \l__tag_graphic_struct_tl {Figure}`

32 `\bool_new:N \l__tag_graphic_artifact_bool`

33 `\bool_new:N \l__tag_graphic_BBox_bool`

34 `\bool_set_true:N \l__tag_graphic_BBox_bool`

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

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

`\l__tag_graphic_cos_fp`

`\l__tag_graphic_scale_fp` 35 `\fp_new:N \l__tag_graphic_sin_fp`

`\l__tag_graphic_lxly_fp` 36 `\fp_new:N \l__tag_graphic_cos_fp`

`\l__tag_graphic_lxuy_fp` 37 `\fp_new:N \l__tag_graphic_lxly_fp`

`\l__tag_graphic_uxly_fp` 38 `\fp_new:N \l__tag_graphic_lxuy_fp`

`\l__tag_graphic_uxuy_fp` 39 `\fp_new:N \l__tag_graphic_uxly_fp`

`\l__tag_graphic_ux_fp` 40 `\fp_new:N \l__tag_graphic_uxuy_fp`

`\l__tag_graphic_ly_fp` 41 `\fp_new:N \l__tag_graphic_ux_fp`

`\l__tag_graphic_lx_fp` 42 `\fp_new:N \l__tag_graphic_ly_fp`

`\l__tag_graphic_uy_fp` 43 `\fp_new:N \l__tag_graphic_lx_fp`

44 `\fp_new:N \l__tag_graphic_uy_fp`

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

`\l__tag_graphic_trim_ly_fp` 45 `\fp_new:N \l__tag_graphic_scale_fp`

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

`\l__tag_graphic_trim_uy_fp` 46 `\fp_new:N \l__tag_graphic_trim_ux_fp`

47 `\fp_new:N \l__tag_graphic_trim_ly_fp`

48 `\fp_new:N \l__tag_graphic_trim_lx_fp`

49 `\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.

```

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

```

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

```

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

```

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

`\Gin@tag@struct@end`

```

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

```

```

96     {
97       \tag_struct_end:
98     }
99     \tag_mc_begin_pop:n{
100   }
101 }

```

(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`.

```

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

```

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

```

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

```

```

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

```

This the main command to calculate the BBox values.

```

152 \Gin@tag@bbox@attribute %new
153 \box\z@

```

and here the tagging stops.

```

154 \Gin@tag@struct@end %new
155 \fi}
156 }

```

#### 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.

```

157 \AddToHook{package/graphicx/after}[latex-lab]
158 {
159     \define@key{Gin}{alt}          {\t1_set:N\l__tag_graphic_alt_t1{\text_purify:n{#1}}}
160     \define@key{Gin}{artifact} []
161     {
162         \bool_set_true:N \l__tag_graphic_artifact_bool
163         \bool_set_false:N \l__tag_graphic_BBox_bool
164     }
165     \define@key{Gin}{actualtext}
166     {
167         \t1_set:N\l__tag_graphic_actual_t1{\text_purify:n{#1}}
168         \bool_set_false:N \l__tag_graphic_BBox_bool
169     }
170     \define@key{Gin}{correct-BBox}
171     {
172         \bool_set_true:N \l__tag_graphic_bboxcorr_bool
173         \seq_set_split:Nnn\l__tag_graphic_bboxcorr_seq{~}{#1~Opt~Opt~Opt~Opt}
174     }
175     \define@key{Gin}{tag}
176     {
177         \str_case:nnF {#1}
178         {
179             {artifact}
180             {
181                 \bool_set_true:N \l__tag_graphic_artifact_bool
182                 \bool_set_false:N \l__tag_graphic_BBox_bool
183             }
184             {false}{\tag_stop:}
185         }

```

```

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

```

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

```

191 \keys_define:nn{tag/picture}
192 {
193     ,alt .code:n =
194     {\tl_set:Nn\l__tag_graphic_alt_tl{\text_purify:n{#1}}}
195     ,artifact .code:n =
196     {
197         \bool_set_true:N \l__tag_graphic_artifact_bool
198         \bool_set_false:N \l__tag_graphic_BBox_bool
199     }
200     ,actualtext .code:n =
201     {
202         \tl_set:Nn\l__tag_graphic_actual_tl{\text_purify:n{#1}}
203         \bool_set_false:N \l__tag_graphic_BBox_bool
204     }
205     ,correct-BBox .code:n =
206     {
207         \bool_set_true:N \l__tag_graphic_bboxcorr_bool
208         \seq_set_split:Nnn\l__tag_graphic_bboxcorr_seq{~}{#1~0pt~0pt~0pt}
209     }
210     ,tag .code:n =
211     {
212         \str_case:nnF {#1}
213         {
214             {artifact}
215             {
216                 \bool_set_true:N \l__tag_graphic_artifact_bool
217                 \bool_set_false:N \l__tag_graphic_BBox_bool
218             }
219             {false}{\tag_stop:}
220         }
221         {\tl_set:Nn\l__tag_graphic_struct_tl{#1}}
222     }
223 }

```

## 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.

```

224 \cs_new_protected:Npn \__tag_graphic_get_trim:
225 {
226     \legacy_if:nTF {Gin@clip}

```

Setting to 0 is not strictly needed but looks cleaner.

```

227     {
228         \fp_zero:N\l__tag_graphic_trim_lx_fp

```



```

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

```

(End of definition for `\__tag_graphic_get_trim:`)

`\__tag_graphic_get_scale:`

```

245 \cs_new_protected:Npn \__tag_graphic_get_scale:
246 {
247 \fp_set:Nn \l__tag_graphic_scale_fp
248 {
249 \str_if_eq:eeTF {\Gin@scalex} { ! }
250 { \Gin@scaley }
251 { \Gin@scalex }
252 }
253 }

```

(End of definition for `\__tag_graphic_get_scale:`)

`\__tag_graphic_applyangle:nmmn`

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 ...

```

254 \cs_new_protected:Npn \__tag_graphic_applyangle:nmmn #1#2#3#4 %lx,ly,ux,uy
255 {
256 \bool_lazy_and:nmT
257 {\cs_if_exist_p:N \Grot@angle }
258 {!\int_compare_p:nNn { \Grot@angle }={0}}
259 {
260 \fp_set:Nn \l__tag_graphic_sin_fp { sind(\Grot@angle) }
261 \fp_set:Nn \l__tag_graphic_cos_fp { cosd(\Grot@angle) }
262 \fp_set:Nn \l__tag_graphic_lx_fp {#1}
263 \fp_set:Nn \l__tag_graphic_ly_fp {#2}
264 \fp_set:Nn \l__tag_graphic_ux_fp {#3}
265 \fp_set:Nn \l__tag_graphic_uy_fp {#4}

```

get the x coordinates (cos,-sin)

```

266 \fp_set:Nn\l__tag_graphic_lxly_fp
267 {
268 -\l__tag_graphic_trim_lx_fp * \l__tag_graphic_cos_fp

```

```

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

```

get the y coordinates (sin,cos)

```
323 \fp_set:Nn\l__tag_graphic_lxly_fp
324 {
325   -\l__tag_graphic_trim_lx_fp * \l__tag_graphic_sin_fp
326   -\l__tag_graphic_trim_ly_fp * \l__tag_graphic_cos_fp
327 }
328 \fp_set:Nn\l__tag_graphic_lxuy_fp
329 {
330   - \l__tag_graphic_trim_lx_fp * \l__tag_graphic_sin_fp
331   +
332   (\l__tag_graphic_uy_fp-\l__tag_graphic_ly_fp-\l__tag_graphic_trim_ly_fp)
333   * \l__tag_graphic_cos_fp
334 }
335 \fp_set:Nn\l__tag_graphic_uxly_fp
336 {
337   (\l__tag_graphic_ux_fp-\l__tag_graphic_lx_fp-\l__tag_graphic_trim_lx_fp)
338   * \l__tag_graphic_sin_fp
339   - \l__tag_graphic_trim_ly_fp * \l__tag_graphic_cos_fp
340 }
341 \fp_set:Nn\l__tag_graphic_uxuy_fp
342 {
343   (\l__tag_graphic_ux_fp-\l__tag_graphic_lx_fp-\l__tag_graphic_trim_lx_fp)
344   * \l__tag_graphic_sin_fp
345   +
346   (\l__tag_graphic_uy_fp-\l__tag_graphic_ly_fp-\l__tag_graphic_trim_ly_fp)
347   * \l__tag_graphic_cos_fp
348 }
349 \tl_gset:Ne\g__tag_graphic_ly_tl
350 {
351   \fp_eval:n
352   {
353     min
354     (
355       \l__tag_graphic_lxly_fp,
356       \l__tag_graphic_lxuy_fp,
357       \l__tag_graphic_uxly_fp,
358       \l__tag_graphic_uxuy_fp
359     )
360     + \l__tag_graphic_ly_fp + \l__tag_graphic_trim_ly_fp
361   }
362 }
363 \tl_gset:Ne\g__tag_graphic_uy_tl
364 {
365   \fp_eval:n
366   {
367     max
368     (
369       \l__tag_graphic_lxly_fp,
370       \l__tag_graphic_lxuy_fp,
371       \l__tag_graphic_uxly_fp,
372       \l__tag_graphic_uxuy_fp,
373     )
374     + \l__tag_graphic_ly_fp + \l__tag_graphic_trim_ly_fp
375   }
}
```

```

376     }
377   }
378 }
379 \cs_generate_variant:Nn\__tag_graphic_applyangle:nmmm {VVVV}

```

(End of definition for `\__tag_graphic_applyangle:nmmm`.)

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

```

380 \cs_new_protected:Npn \__tag_graphic_applycorr:NNNN #1 #2 #3 #4
381 {
382   \bool_if:NT\l__tag_graphic_bboxcorr_bool
383   {
384     \tl_set:Ne #1
385     {
386       \fp_eval:n
387       {
388         #1
389         +
390         \dim_to_decimal_in_bp:n {\seq_item:Nn \l__tag_graphic_bboxcorr_seq {1} }
391       }
392     }
393     \tl_set:Ne #2
394     {
395       \fp_eval:n
396       {
397         #2
398         +
399         \dim_to_decimal_in_bp:n {\seq_item:Nn \l__tag_graphic_bboxcorr_seq {2} }
400       }
401     }
402     \tl_set:Ne #3
403     {
404       \fp_eval:n
405       {
406         #3
407         +
408         \dim_to_decimal_in_bp:n {\seq_item:Nn \l__tag_graphic_bboxcorr_seq {3} }
409       }
410     }
411     \tl_set:Ne #4
412     {
413       \fp_eval:n
414       {
415         #4
416         +
417         \dim_to_decimal_in_bp:n {\seq_item:Nn \l__tag_graphic_bboxcorr_seq {4} }
418       }
419     }
420   }
421 }

```

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

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

```

422 \cs_new_protected:Npn \Gin@tag@bbox@attribute
423   {

```

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

```

424   \bool_lazy_all:nT
425     {
426       {\tag_if_active_p:}
427       {\!\l__tag_graphic_artifact_bool}
428       {\!\l__tag_graphic_BBox_bool}
429     }
430     {
431       \__tag_graphic_get_scale:
432       \__tag_graphic_get_trim:
433       \int_gincr:N\g__tag_graphic_int
434       \tl_set:N\l__tag_graphic_currentlabel_tl {\__tag_graphic_\int_use:N \g__tag_graphic_int
435       \__tag_graphic_savepos:e { \l__tag_graphic_currentlabel_tl }
436       \tl_gset:N\g__tag_graphic_lx_tl
437         {
438           \dim_to_decimal_in_bp:n
439           { \property_ref:een {\l__tag_graphic_currentlabel_tl}{xpos}{0}sp }
440         }
441       \tl_gset:N\g__tag_graphic_ly_tl
442         {
443           \dim_to_decimal_in_bp:n
444           { \property_ref:een {\l__tag_graphic_currentlabel_tl}{ypos}{0}sp }
445         }
446       \tl_gset:N\g__tag_graphic_ux_tl
447         {
448           \fp_eval:n
449           {
450             \g__tag_graphic_lx_tl
451             +
452             \dim_to_decimal_in_bp:n { \Gin@req@width }
453           }
454         }
455       \tl_gset:N\g__tag_graphic_uy_tl
456         {
457           \fp_eval:n
458           {
459             \g__tag_graphic_ly_tl
460             +
461             \dim_to_decimal_in_bp:n { \Gin@req@height }
462           }
463         }

```

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

```

464       \legacy_if:nF {\Gin@clip}
465       {
466         \tl_gset:N\g__tag_graphic_ux_tl
467           {
468             \fp_eval:n
469             {
470               \g__tag_graphic_ux_tl
471               +
472               \l__tag_graphic_trim_ux_fp

```

```

473     }
474   }
475   \tl_gset:Ne\g__tag_graphic_lx_tl
476   {
477     \fp_eval:n
478     {
479       \g__tag_graphic_lx_tl
480       -
481       \l__tag_graphic_trim_lx_fp
482     }
483   }
484   \tl_gset:Ne\g__tag_graphic_uy_tl
485   {
486     \fp_eval:n
487     {
488       \g__tag_graphic_uy_tl
489       +
490       \l__tag_graphic_trim_uy_fp
491     }
492   }
493   \tl_gset:Ne\g__tag_graphic_ly_tl
494   {
495     \fp_eval:n
496     {
497       \g__tag_graphic_ly_tl
498       -
499       \l__tag_graphic_trim_ly_fp
500     }
501   }
502 }

```

If there is an angle we now rotate the values.

```

503   \__tag_graphic_applyangle:VVVV
504   \g__tag_graphic_lx_tl
505   \g__tag_graphic_ly_tl
506   \g__tag_graphic_ux_tl
507   \g__tag_graphic_uy_tl

```

At last we have to add the correction values

```

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

```

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.

```

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

```

*(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.

```

539 \newcommand\picture@tag@bbox@attribute
540 {
541     \bool_lazy_all:nT
542     {
543         {\tag_if_active_p:}
544         {\!\__tag_graphic_artifact_bool}
545         {\!\__tag_graphic_BBox_bool}
546     }
547     {
548         \int_gincr:N\g__tag_graphic_int
549         \tl_set:N\l__tag_graphic_currentlabel_tl {\__tag_graphic_\int_use:N \g__tag_graphic_int}
550         \__tag_graphic_savepos:e { \l__tag_graphic_currentlabel_tl }
551         \tl_gset:N\g__tag_graphic_lx_tl
552         {
553             \dim_to_decimal_in_bp:n
554             { \property_ref:een {\l__tag_graphic_currentlabel_tl}{xpos}{0}sp }
555         }
556         \tl_gset:N\g__tag_graphic_ly_tl
557         {
558             \dim_to_decimal_in_bp:n
559             { \property_ref:een {\l__tag_graphic_currentlabel_tl}{ypos}{0}sp - \dp\@picbox }
560         }
561         \tl_gset:N\g__tag_graphic_ux_tl
562         {
563             \dim_to_decimal_in_bp:n
564             {
565                 \g__tag_graphic_lx_tl bp + \wd\@picbox

```

```

566     }
567   }
568   \tl_gset:Ne \g__tag_graphic_uy_tl
569   {
570     \dim_to_decimal_in_bp:n
571     {
572       \g__tag_graphic_ly_tl bp + \ht\@picbox + \dp\@picbox
573     }
574   }
575   \__tag_graphic_applycorr:NNNN
576   \g__tag_graphic_lx_tl
577   \g__tag_graphic_ly_tl
578   \g__tag_graphic_ux_tl
579   \g__tag_graphic_uy_tl
580   \bool_if:NT\l__tag_graphic_debug_bool
581   {
582     \__tag_graphic_show_bbox:VVVNe
583     \g__tag_graphic_lx_tl
584     \g__tag_graphic_ly_tl
585     \g__tag_graphic_ux_tl
586     \g__tag_graphic_uy_tl
587     {red}
588     {\int_use:N\g__tag_graphic_int}
589   }
590   \__tag_prop_gput:cnx
591   { g__tag_struct_\int_eval:n {\c@g__tag_struct_abs_int}_prop }
592   { A }
593   {
594     <<
595     /O /Layout /BBox~
596     [
597       \g__tag_graphic_lx_tl\c_space_tl
598       \g__tag_graphic_ly_tl\c_space_tl
599       \g__tag_graphic_ux_tl\c_space_tl
600       \g__tag_graphic_uy_tl
601     ]
602     >>
603   }
604 }
605 }
606

```

(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.

```

607 \RenewDocumentCommand\picture{0{m}
608 {
609   \leavevmode
610   \keys_set:nn{tag/picture}{#1} %
611   \tl_set:Nn\l__tag_graphic_alt_dflt_tl {picture-environment}
612   \pictur@#2
613 }

```



inside the picture box we stop tagging.

```

614 \def\@picture(#1,#2)(#3,#4){%
615   \@defaultunitsset\@picht{#2}\unitlength
616   \@defaultunitsset\@tempdimc{#1}\unitlength
617   \Gin@tag@struct@begin
618   \setbox\@picbox\hb@xt@\@tempdimc\bgroup
619   \tag_stop: %do not tag inside the picture box
620   \@defaultunitsset\@tempdimc{#3}\unitlength
621   \hskip -\@tempdimc
622   \@defaultunitsset\@tempdimc{#4}\unitlength
623   \lower\@tempdimc\hbox\bgroup
624   \ignorespaces}

625 \def\endpicture{%
626   \egroup\hss\egroup
627   \ht\@picbox\@picht\dp\@picbox\z@
628   \picture@tag@bbox@attribute
629   \mbox{\box\@picbox}
630   \Gin@tag@struct@end}

```

## 4.7 Debugging code

`\_tag_graphic_show_bbox:nnnnn`

```

631 \cs_new_protected:Npn \_tag_graphic_show_bbox:nnnnn #1#2#3#4#5#6%#5 color, #6 graphic
632 {
633   \iow_log:n {tag/graphic~debug:~BBox~of~graphics~#6~is~#1~#2~#3~#4}
634   \hook_gput_code:nnn
635   {shipout/foreground}
636   {tag/graphic}
637   {
638     \int_compare:nNnT
639     {\g_shipout_readonly_int}
640     =
641     {\property_ref:een{\_tag_graphic_#6}{abspage}{0}}
642     {
643       \put
644       (#1 bp,\dim_eval:n{-\paperheight + \dim_eval:n{#2 bp}})
645       {
646         \opacity_select:n{0.5}\color_select:n{#5}
647         \rule
648         {\dim_eval:n {#3 bp-\dim_eval:n{#1 bp}}}
649         {\dim_eval:n {#4 bp-\dim_eval:n{#2 bp}}}
650       }
651     }
652   }
653 }
654 \cs_generate_variant:Nn \_tag_graphic_show_bbox:nnnnn {VVVVne}

```

*(End of definition for `\_tag_graphic_show_bbox:nnnnn`.)*

```

655 </package>
656 <*latex-lab>
657 \ProvidesFile{graphic-latex-lab-testphase.ltx}
658 [\ltxlabgraphicdate\space v\ltxlabgraphicversion\space latex-lab wrapper graphic]

```

```
659 \RequirePackage{latex-lab-testphase-graphic}  
660 \end{document}
```