The latex-lab-graphic package Tagging of included graphics

LaTeX 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 `\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 completly. It is then the responsability 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.

```
⟨@@=tag⟩
⟨*package⟩
```

4 Implementation

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

We load l3opacity for the debug code

```
\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.

```
\cs_new_protected:Npn\__tag_graphic_savepos:n #1
 {
  \tex_savepos:D
  \property_record:nn{#1}{xpos,ypos,abspage}
  \tex_savepos:D
 }
\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

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

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

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

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

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

\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

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 ...

```
\tl_new:N \g__tag_graphic_lx_tl
\tl_new:N \g__tag_graphic_ly_tl
\tl_new:N \g__tag_graphic_ux_tl
\tl_new:N \g__tag_graphic_uy_tl
\seq_new:N\l__tag_graphic_bboxcorr_seq
\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.

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

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

\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

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

```
\tl_new:N \l__tag_graphic_alt_tl
\tl_new:N \l__tag_graphic_alt_dflt_tl
\tl_set:Nn \l__tag_graphic_alt_dflt_tl {\Gin@base\Gin@ext}
\tl_new:N \l__tag_graphic_actual_tl
\tl_new:N \l__tag_graphic_struct_tl
\tl_set:Nn\l__tag_graphic_struct_tl {Figure}
\bool_new:N\l__tag_graphic_artifact_bool
\bool_new:N\l__tag_graphic_BBox_bool
\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
\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

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

```
\fp_new:N\l__tag_graphic_sin_fp
\fp_new:N\l__tag_graphic_cos_fp
\fp_new:N\l__tag_graphic_lxly_fp
\fp_new:N\l__tag_graphic_lxuy_fp
\fp_new:N\l__tag_graphic_uxly_fp
\fp_new:N\l__tag_graphic_uxuy_fp
\fp_new:N\l__tag_graphic_ux_fp
\fp_new:N\l__tag_graphic_ly_fp
\fp_new:N\l__tag_graphic_lx_fp
\fp_new:N\l__tag_graphic_uy_fp
```

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

```
\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.

```
\fp_new:N\l__tag_graphic_trim_ux_fp
\fp_new:N\l__tag_graphic_trim_ly_fp
\fp_new:N\l__tag_graphic_trim_lx_fp
\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.

```
\msg_new:nnn {tag}{alt-text-missing}
  {
    Alternative~text~for~graphic~is~missing.\\
    Using~’#1’~instead
  }
\cs_new_protected:Npn\Gin@tag@struct@begin
 {
   \tag_if_active:T
    {
      \tag_mc_end_push:
```

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

```
      \bool_if:NTF\l__tag_graphic_artifact_bool
       {
         \tag_mc_begin:n{artifact}
       }
       {
         \tl_if_empty:NTF\l__tag_graphic_actual_tl
           {
             \tl_if_empty:NT\l__tag_graphic_alt_tl
               {
                 \msg_warning:nne{tag}{alt-text-missing}{\l__tag_graphic_alt_dflt_tl}
                 \tl_set:Ne\l__tag_graphic_alt_tl {\l__tag_graphic_alt_dflt_tl}
               }
             \tag_struct_begin:n
               {
                tag=\l__tag_graphic_struct_tl,
                alt=\l__tag_graphic_alt_tl,
               }
           }
           {
             \tag_struct_begin:n
               {
                 tag=Span,
                 actualtext=\l__tag_graphic_actual_tl,
               }
             \bool_set_false:N\l__tag_graphic_BBox_bool
           }
        \tag_mc_begin:n{}
       }
    }
 }
```

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

\Gin@tag@struct@end

```
\cs_new_protected:Npn\Gin@tag@struct@end
 {
   \tag_if_active:T
    {
     \tag_mc_end:
     \bool_if:NF\l__tag_graphic_artifact_bool
```

```
      {
        \tag_struct_end:
      }
     \tag_mc_begin_pop:n{}
    }
}
```

(*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`.

```
\AddToHook{package/graphics/after}
 {
  \def\Gin@setfile#1#2#3{%
  \ifx\\#2\\\Gread@false\fi
  \ifGin@bbox\else
    \ifGread@
      \csname Gread@%
         \expandafter\ifx\csname Gread@#1\endcsname\relax
           eps%
         \else
           #1%
         \fi
      \endcsname{\Gin@base#2}%
    \else
      \Gin@nosize{#3}%
    \fi
  \fi
  \Gin@viewport@code
  \Gin@nat@height\Gin@ury bp%
  \advance\Gin@nat@height-\Gin@lly bp%
  \Gin@nat@width\Gin@urx bp%
  \advance\Gin@nat@width-\Gin@llx bp%
  \Gin@req@sizes
  \expandafter\ifx\csname Ginclude@#1\endcsname\relax
    \Gin@drafttrue
    \expandafter\ifx\csname Gread@#1\endcsname\relax
      \@latex@error{Can not include graphics of type: #1}\@ehc
      \global\expandafter\let\csname Gread@#1\endcsname\@empty
    \fi
  \fi
  \leavevmode
```

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

```
  \Gin@tag@struct@begin %NEW
  \ifGin@draft
      \hb@xt@\Gin@req@width{%
         \vrule\hss
         \vbox to \Gin@req@height{%
            \hrule \@width \Gin@req@width
            \vss
            \edef\@tempa{#3}%
```

```
          \rlap{ \ttfamily\expandafter\strip@prefix\meaning\@tempa}%
          \vss
          \hrule}%
       \hss\vrule}%
  \else
    \@addtofilelist{#3}%
    \ProvidesFile{#3}[Graphic file (type #1)]%
    \setbox\z@\hbox{\csname Ginclude@#1\endcsname{#3}}%
    \dp\z@\z@
    \ht\z@\Gin@req@height
    \wd\z@\Gin@req@width
```

This the main command to calculate the BBox values.

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

and here the tagging stops.

```
  \Gin@tag@struct@end %new
  \fi}
 }
```

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.

```
\AddToHook{package/graphicx/after}[latex-lab]
  {
    \define@key{Gin}{alt}       {\tl_set:Ne\l__tag_graphic_alt_tl{\text_purify:n{#1}}}
    \define@key{Gin}{artifact}[]
      {
        \bool_set_true:N \l__tag_graphic_artifact_bool
        \bool_set_false:N \l__tag_graphic_BBox_bool
      }
    \define@key{Gin}{actualtext}
     {
       \tl_set:Ne\l__tag_graphic_actual_tl{\text_purify:n{#1}}
       \bool_set_false:N \l__tag_graphic_BBox_bool
     }
    \define@key{Gin}{correct-BBox}
      {
        \bool_set_true:N \l__tag_graphic_bboxcorr_bool
        \seq_set_split:Nnn\l__tag_graphic_bboxcorr_seq{~}{#1~0pt~0pt~0pt~0pt}
      }
    \define@key{Gin}{tag}
      {
        \str_case:nnF {#1}
          {
            {artifact}
            {
               \bool_set_true:N \l__tag_graphic_artifact_bool
               \bool_set_false:N \l__tag_graphic_BBox_bool
            }
            {false}{\tag_stop:}
          }
          {\tl_set:Nn\l__tag_graphic_struct_tl{#1}}
      }
  }
\AddToHook{package/graphics/after}[latex-lab]
  {\RequirePackage{graphicx}}
```

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

```
\keys_define:nn{tag/picture}
   {
     ,alt .code:n =
        {\tl_set:Ne\l__tag_graphic_alt_tl{\text_purify:n{#1}}}
     ,artifact .code:n =
        {
          \bool_set_true:N \l__tag_graphic_artifact_bool
          \bool_set_false:N \l__tag_graphic_BBox_bool
        }
     ,actualtext .code:n =
        {
          \tl_set:Ne\l__tag_graphic_actual_tl{\text_purify:n{#1}}
          \bool_set_false:N \l__tag_graphic_BBox_bool
        }
     ,correct-BBox .code:n =
      {
        \bool_set_true:N \l__tag_graphic_bboxcorr_bool
        \seq_set_split:Nnn\l__tag_graphic_bboxcorr_seq{~}{#1~0pt~0pt~0pt~0pt}
      }
    ,tag .code:n =
      {
        \str_case:nnF {#1}
          {
            {artifact}
            {
               \bool_set_true:N \l__tag_graphic_artifact_bool
               \bool_set_false:N \l__tag_graphic_BBox_bool
            }
            {false}{\tag_stop:}
          }
          {\tl_set:Nn\l__tag_graphic_struct_tl{#1}}
      }
   }
```

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.

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

Setting to 0 is not strictly needed but looks cleaner.

```
    {
      \fp_zero:N\l__tag_graphic_trim_lx_fp
      \fp_zero:N\l__tag_graphic_trim_ly_fp
      \fp_zero:N\l__tag_graphic_trim_ux_fp
      \fp_zero:N\l__tag_graphic_trim_uy_fp
    }
    {
      \fp_set:Nn \l__tag_graphic_trim_lx_fp {\l__tag_graphic_scale_fp*\Gin@vllx}
      \fp_set:Nn \l__tag_graphic_trim_ly_fp {\l__tag_graphic_scale_fp*\Gin@vlly}
      \fp_set:Nn \l__tag_graphic_trim_ux_fp {\l__tag_graphic_scale_fp*\Gin@vurx}
      \fp_set:Nn \l__tag_graphic_trim_uy_fp {\l__tag_graphic_scale_fp*\Gin@vury}
      \cs_if_exist:NT \Gin@ollx
        {
         \fp_set:Nn \l__tag_graphic_trim_ux_fp {\l__tag_graphic_scale_fp* (\Gin@ourx-(\Gin@ur
         \fp_set:Nn \l__tag_graphic_trim_uy_fp {\l__tag_graphic_scale_fp* (\Gin@oury-(\Gin@ur
        }
    }
 }
```

(*End of definition for* `\__tag_graphic_get_trim:`.)

__tag_graphic_get_scale:

```
\cs_new_protected:Npn \__tag_graphic_get_scale:
 {
   \fp_set:Nn \l__tag_graphic_scale_fp
     {
       \str_if_eq:eeTF {\Gin@scalex} { ! }
        { \Gin@scaley }
        { \Gin@scalex }
     }
 }
```

(*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 ...

```
\cs_new_protected:Npn \__tag_graphic_applyangle:nnnn #1#2#3#4 %lx,ly,ux,uy
 {
   \bool_lazy_and:nnT
     {\cs_if_exist_p:N \Grot@angle }
     {! \int_compare_p:nNn { \Grot@angle }={0}}
     {
       \fp_set:Nn \l__tag_graphic_sin_fp  { sind(\Grot@angle) }
       \fp_set:Nn \l__tag_graphic_cos_fp  { cosd(\Grot@angle) }
       \fp_set:Nn \l__tag_graphic_lx_fp {#1}
       \fp_set:Nn \l__tag_graphic_ly_fp {#2}
       \fp_set:Nn \l__tag_graphic_ux_fp {#3}
       \fp_set:Nn \l__tag_graphic_uy_fp {#4}
```

get the x coordinates (cos,-sin)

```
       \fp_set:Nn\l__tag_graphic_lxly_fp
        {
          -\l__tag_graphic_trim_lx_fp *  \l__tag_graphic_cos_fp
```

```
         +\l__tag_graphic_trim_ly_fp *  \l__tag_graphic_sin_fp
       }
      \fp_set:Nn\l__tag_graphic_lxuy_fp
       {
         (-\l__tag_graphic_trim_lx_fp) * \l__tag_graphic_cos_fp
         +
          (\l__tag_graphic_uy_fp-\l__tag_graphic_ly_fp-\l__tag_graphic_trim_ly_fp)
           * (-\l__tag_graphic_sin_fp)
       }
      \fp_set:Nn\l__tag_graphic_uxly_fp
       {
         (\l__tag_graphic_ux_fp-\l__tag_graphic_lx_fp-\l__tag_graphic_trim_lx_fp)
          * \l__tag_graphic_cos_fp
         +
         (\l__tag_graphic_trim_ly_fp) * (\l__tag_graphic_sin_fp)
       }
      \fp_set:Nn\l__tag_graphic_uxuy_fp
       {
         (\l__tag_graphic_ux_fp-\l__tag_graphic_lx_fp-\l__tag_graphic_trim_lx_fp)
           *  \l__tag_graphic_cos_fp
        +
         (\l__tag_graphic_uy_fp-\l__tag_graphic_ly_fp-\l__tag_graphic_trim_ly_fp)
           * (-\l__tag_graphic_sin_fp)
       }
      \tl_gset:Ne\g__tag_graphic_lx_tl
        {
         \fp_eval:n
           {
             min
              (
                \l__tag_graphic_lxly_fp,
                \l__tag_graphic_lxuy_fp,
                \l__tag_graphic_uxly_fp,
                \l__tag_graphic_uxuy_fp,
              )
              +\l__tag_graphic_lx_fp
              +\l__tag_graphic_trim_lx_fp
           }
        }
      \tl_gset:Ne\g__tag_graphic_ux_tl
        {
          \fp_eval:n
            {
              max
               (
                 \l__tag_graphic_lxly_fp,
                 \l__tag_graphic_lxuy_fp,
                 \l__tag_graphic_uxly_fp,
                 \l__tag_graphic_uxuy_fp
               )
               +\l__tag_graphic_lx_fp
               +\l__tag_graphic_trim_lx_fp
            }
        }
```

get the y coordinates (sin,cos)

```
    \fp_set:Nn\l__tag_graphic_lxly_fp
     {
        -\l__tag_graphic_trim_lx_fp *  \l__tag_graphic_sin_fp
        -\l__tag_graphic_trim_ly_fp *  \l__tag_graphic_cos_fp
     }
    \fp_set:Nn\l__tag_graphic_lxuy_fp
     {
       - \l__tag_graphic_trim_lx_fp *  \l__tag_graphic_sin_fp
       +
       (\l__tag_graphic_uy_fp-\l__tag_graphic_ly_fp-\l__tag_graphic_trim_ly_fp)
         * \l__tag_graphic_cos_fp
     }
    \fp_set:Nn\l__tag_graphic_uxly_fp
     {
       (\l__tag_graphic_ux_fp-\l__tag_graphic_lx_fp-\l__tag_graphic_trim_lx_fp)
         * \l__tag_graphic_sin_fp
       - \l__tag_graphic_trim_ly_fp * \l__tag_graphic_cos_fp
     }
    \fp_set:Nn\l__tag_graphic_uxuy_fp
     {
       (\l__tag_graphic_ux_fp-\l__tag_graphic_lx_fp-\l__tag_graphic_trim_lx_fp)
         * \l__tag_graphic_sin_fp
      +
       (\l__tag_graphic_uy_fp-\l__tag_graphic_ly_fp-\l__tag_graphic_trim_ly_fp)
         * \l__tag_graphic_cos_fp
     }
    \tl_gset:Ne\g__tag_graphic_ly_tl
      {
       \fp_eval:n
         {
           min
            (
              \l__tag_graphic_lxly_fp,
              \l__tag_graphic_lxuy_fp,
              \l__tag_graphic_uxly_fp,
              \l__tag_graphic_uxuy_fp
            )
           + \l__tag_graphic_ly_fp + \l__tag_graphic_trim_ly_fp
         }
      }
    \tl_gset:Ne\g__tag_graphic_uy_tl
      {
        \fp_eval:n
          {
            max
             (
               \l__tag_graphic_lxly_fp,
               \l__tag_graphic_lxuy_fp,
               \l__tag_graphic_uxly_fp,
               \l__tag_graphic_uxuy_fp,
             )
            + \l__tag_graphic_ly_fp + \l__tag_graphic_trim_ly_fp
          }
         }
     }
 }
\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 ...

```
\cs_new_protected:Npn \__tag_graphic_applycorr:NNNN #1 #2 #3 #4
 {
  \bool_if:NT\l__tag_graphic_bboxcorr_bool
    {
     \tl_set:Ne #1
       {
         \fp_eval:n
          {
           #1
           +
           \dim_to_decimal_in_bp:n {\seq_item:Nn \l__tag_graphic_bboxcorr_seq {1} }
          }
       }
     \tl_set:Ne #2
       {
         \fp_eval:n
          {
           #2
           +
           \dim_to_decimal_in_bp:n {\seq_item:Nn \l__tag_graphic_bboxcorr_seq {2} }
          }
       }
     \tl_set:Ne #3
       {
         \fp_eval:n
          {
           #3
           +
           \dim_to_decimal_in_bp:n {\seq_item:Nn \l__tag_graphic_bboxcorr_seq {3} }
          }
       }
     \tl_set:Ne #4
       {
         \fp_eval:n
          {
           #4
           +
           \dim_to_decimal_in_bp:n {\seq_item:Nn \l__tag_graphic_bboxcorr_seq {4} }
          }
       }
     }
  }
```

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

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

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

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

```
  \bool_lazy_all:nT
    {
      {\tag_if_active_p:}
      {!\l__tag_graphic_artifact_bool}
      {\l__tag_graphic_BBox_bool}
    }
    {
      \__tag_graphic_get_scale:
      \__tag_graphic_get_trim:
      \int_gincr:N\g__tag_graphic_int
      \tl_set:Ne\l__tag_graphic_currentlabel_tl {__tag_graphic_\int_use:N \g__tag_graphic_int
      \__tag_graphic_savepos:e { \l__tag_graphic_currentlabel_tl }
      \tl_gset:Ne\g__tag_graphic_lx_tl
        {
          \dim_to_decimal_in_bp:n
            { \property_ref:een {\l__tag_graphic_currentlabel_tl}{xpos}{0}sp }
        }
      \tl_gset:Ne\g__tag_graphic_ly_tl
        {
          \dim_to_decimal_in_bp:n
            { \property_ref:een {\l__tag_graphic_currentlabel_tl}{ypos}{0}sp }
        }
      \tl_gset:Ne\g__tag_graphic_ux_tl
          {
            \fp_eval:n
             {
               \g__tag_graphic_lx_tl
               +
               \dim_to_decimal_in_bp:n { \Gin@req@width }
             }
          }
       \tl_gset:Ne\g__tag_graphic_uy_tl
          {
            \fp_eval:n
             {
               \g__tag_graphic_ly_tl
               +
                \dim_to_decimal_in_bp:n  { \Gin@req@height }
             }
          }
```

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

```
      \legacy_if:nF {Gin@clip}
        {
          \tl_gset:Ne\g__tag_graphic_ux_tl
            {
              \fp_eval:n
                {
                  \g__tag_graphic_ux_tl
                  +
                  \l__tag_graphic_trim_ux_fp
            }
        }
      \tl_gset:Ne\g__tag_graphic_lx_tl
        {
          \fp_eval:n
            {
              \g__tag_graphic_lx_tl
               -
              \l__tag_graphic_trim_lx_fp
            }
        }
      \tl_gset:Ne\g__tag_graphic_uy_tl
        {
          \fp_eval:n
            {
              \g__tag_graphic_uy_tl
              +
              \l__tag_graphic_trim_uy_fp
            }
        }
      \tl_gset:Ne\g__tag_graphic_ly_tl
        {
          \fp_eval:n
            {
              \g__tag_graphic_ly_tl
              -
              \l__tag_graphic_trim_ly_fp
            }
        }
    }
```

If there is an angle we now rotate the values.

```
  \__tag_graphic_applyangle:VVVV
    \g__tag_graphic_lx_tl
    \g__tag_graphic_ly_tl
    \g__tag_graphic_ux_tl
    \g__tag_graphic_uy_tl
```

At last we have to add the correction values

```
  \__tag_graphic_applycorr:NNNN
    \g__tag_graphic_lx_tl
    \g__tag_graphic_ly_tl
    \g__tag_graphic_ux_tl
    \g__tag_graphic_uy_tl

\bool_if:NT\l__tag_graphic_debug_bool
 {
   \__tag_graphic_show_bbox:VVVVne
    \g__tag_graphic_lx_tl
    \g__tag_graphic_ly_tl
    \g__tag_graphic_ux_tl
    \g__tag_graphic_uy_tl
    {red}
    {\int_use:N\g__tag_graphic_int}
 }
```

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.

```
      \__tag_prop_gput:cnx
         { g__tag_struct_\int_eval:n {\c@g__tag_struct_abs_int}_prop }
         { A }
         {
           <<
             /O /Layout /BBox~
             [
              \g__tag_graphic_lx_tl\c_space_tl
              \g__tag_graphic_ly_tl\c_space_tl
              \g__tag_graphic_ux_tl\c_space_tl
              \g__tag_graphic_uy_tl
             ]
           >>
         }
    }
  }
```

(*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.

```
\newcommand\picture@tag@bbox@attribute
 {
   \bool_lazy_all:nT
    {
      {\tag_if_active_p:}
      {!\l__tag_graphic_artifact_bool}
      {\l__tag_graphic_BBox_bool}
    }
    {
      \int_gincr:N\g__tag_graphic_int
      \tl_set:Ne\l__tag_graphic_currentlabel_tl {__tag_graphic_\int_use:N \g__tag_graphic_int
      \__tag_graphic_savepos:e { \l__tag_graphic_currentlabel_tl }
      \tl_gset:Ne \g__tag_graphic_lx_tl
        {
          \dim_to_decimal_in_bp:n
          { \property_ref:een {\l__tag_graphic_currentlabel_tl}{xpos}{0}sp  }
        }
      \tl_gset:Ne \g__tag_graphic_ly_tl
        {
          \dim_to_decimal_in_bp:n
          { \property_ref:een {\l__tag_graphic_currentlabel_tl}{ypos}{0}sp - \dp\@picbox }
        }
      \tl_gset:Ne \g__tag_graphic_ux_tl
        {
          \dim_to_decimal_in_bp:n
           {
             \g__tag_graphic_lx_tl bp + \wd\@picbox
           }
        }
      \tl_gset:Ne \g__tag_graphic_uy_tl
        {
          \dim_to_decimal_in_bp:n
           {
             \g__tag_graphic_ly_tl bp + \ht\@picbox + \dp\@picbox
           }
        }
      \__tag_graphic_applycorr:NNNN
           \g__tag_graphic_lx_tl
           \g__tag_graphic_ly_tl
           \g__tag_graphic_ux_tl
           \g__tag_graphic_uy_tl
      \bool_if:NT\l__tag_graphic_debug_bool
        {
          \__tag_graphic_show_bbox:VVVVne
           \g__tag_graphic_lx_tl
           \g__tag_graphic_ly_tl
           \g__tag_graphic_ux_tl
           \g__tag_graphic_uy_tl
           {red}
           {\int_use:N\g__tag_graphic_int}
        }
      \__tag_prop_gput:cnx
            { g__tag_struct_\int_eval:n {\c@g__tag_struct_abs_int}_prop }
            { A }
            {
              <<
                /O /Layout /BBox~
                [
                 \g__tag_graphic_lx_tl\c_space_tl
                 \g__tag_graphic_ly_tl\c_space_tl
                 \g__tag_graphic_ux_tl\c_space_tl
                 \g__tag_graphic_uy_tl
                ]
              >>
           }
     }
  }

```

(*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.

```
\RenewDocumentCommand\picture{O{}m}
 {
   \leavevmode
   \keys_set:nn{tag/picture}{#1} %
   \tl_set:Nn\l__tag_graphic_alt_dflt_tl {picture~environment}
   \pictur@#2
 }
```

inside the picture box we stop tagging.

```
\def\@picture(#1,#2)(#3,#4){%
  \@defaultunitsset\@picht{#2}\unitlength
  \@defaultunitsset\@tempdimc{#1}\unitlength
  \Gin@tag@struct@begin
  \setbox\@picbox\hb@xt@\@tempdimc\bgroup
    \tag_stop: %do not tag inside the picture box
    \@defaultunitsset\@tempdimc{#3}\unitlength
    \hskip -\@tempdimc
    \@defaultunitsset\@tempdimc{#4}\unitlength
    \lower\@tempdimc\hbox\bgroup
      \ignorespaces}
```

```
\def\endpicture{%
  \egroup\hss\egroup
    \ht\@picbox\@picht\dp\@picbox\z@
    \picture@tag@bbox@attribute
    \mbox{\box\@picbox}
    \Gin@tag@struct@end}
```

4.7 Debugging code

__tag_graphic_show_bbox:nnnnnn

```
\cs_new_protected:Npn \__tag_graphic_show_bbox:nnnnnn #1#2#3#4#5#6%#5 color, #6 graphic
 {
    \iow_log:n {tag/graphic~debug:~BBox~of~graphics~#6~is~#1~#2~#3~#4}
    \hook_gput_code:nnn
     {shipout/foreground}
     {tag/graphic}
     {
      \int_compare:nNnT
       {\g_shipout_readonly_int}
       =
       {\property_ref:een{__tag_graphic_#6}{abspage}{0}}
       {
        \put
         (#1 bp,\dim_eval:n{-\paperheight + \dim_eval:n{#2 bp}})
         {
          \opacity_select:n{0.5}\color_select:n{#5}
          \rule
           {\dim_eval:n {#3 bp-\dim_eval:n{#1 bp}}}
           {\dim_eval:n {#4 bp-\dim_eval:n{#2 bp}}}
         }
       }
     }
 }
 \cs_generate_variant:Nn  \__tag_graphic_show_bbox:nnnnnn {VVVVne}
```

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

```
⟨/package⟩
```

```
⟨*latex-lab⟩
\ProvidesFile{graphic-latex-lab-testphase.ltx}
        [\ltlabgraphicdate\space v\ltlabgraphicversion\space latex-lab wrapper graphic]
```

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