Current typesetting position in pdfTeX

Vít Zýka

Abstract

Among the relatively little-known pdfTeX extensions is a possibility of obtaining the current typesetting position. It can be used later for placing objects. This article gives a description of related primitives and shows an example of usage.

1 Motivation

TeX proceeds sequentially when building a page. It places an object (box, char, rule, space) next to another one and it shifts the current typesetting point. If we need to stack up several objects (e.g. color background, stamps) we have to proceed in four steps: 1. remember a current position, 2. move to the required position, 3. place the object here, and 4. return back to the starting position. The current typesetting point must not be influenced when processing these four steps. This does not limit us in most cases but it is a bit unwieldy.

We face a worse case when placing/scaling of the object depends on two or even more points. Imagine a color background or a frame surrounding several paragraphs with nonzero vertical stretchability between, or drawing a diagram with an arrow between cells, or connecting two words inside a paragraph by line. For all these applications we need to remember the typesetting points for later drawing of the dependent object. Classical TeX has no instrument for these tasks. The only way used to be a cooperation with a specific driver (e.g. via PostScript operators), which makes the document driver-dependent. PdfTeX has a more straightforward and portable solution — it provides new primitives for obtaining a current typesetting position.

2 New primitives

There are three new primitives for working with the current position. The first is `\pdfsavepos`, which saves a mark in the main vertical list. After the page formatting, during `\shipout` operation, the mark is processed and the absolute typesetting position is saved in relation to the left bottom page corner. Afterwards, this (x, y) position can be read by primitives `\pdflastxpos` and `\pdflastypos`. Each of them returns an integer value representing the distance in scaled points (`sp`).

Since `\pdfsavepos` is processed at `\shipout` time, when typesetting is done, we need to write the position values to a file, then read and use them in the next TeX run.

This usage is not simple and therefore we will show it in an example. Our goal will be to connect two words inside a paragraph by a line.

3 Example: Drawing a line inside a paragraph

Let us solve this task: *draw a line from place A to place B*. The places are to be marked by writing `\posMark{place_label}` anywhere on a single page, even inside a paragraph. The two marks could be used to draw a line between them. The example illustrating the idea is shown inside this paragraph. It was typeset by the following code:

```
Let us solve this task:
{\it draw a line\posMark{A} ...
... to draw a line\posMark{B} between ...
```

How does it work? Let us start our description with helping macros.[1] First we need to draw a line. Let's make that line be gray and 2 bp wide. Low level PDF or PostScript operators solve this simple drawing, so we can avoid loading large vector drawing packages such as Ti*k*Z or PSTricks. The main output drivers pdfTeX and Dvips are distinguished by the `\ifpdf` macro:

```
\ifpdf
  % 1,2=start x y; 3,4=stop x y <bp unit>
  \def\Line#1,#2--#3,#4{%
    \pdfliteral page
      {0.7 G 2 w #1 #2 m #3 #4 l S }}
\else
  \def\Line#1,#2--#3,#4{%
    \special{" 0.7 setgray 2 setlinewidth
      #1 #2 moveto #3 #4 lineto stroke }}
\fi
```

The next macro cuts off the unit *pt* from a dimension:

```
{\catcode`\p=12 \catcode`\t=12
 \gdef\removePT#1pt{#1}}
```

Our last helping action is conversion from *sp* units to *bp*, which is a base unit in PDF or PostScript:

```
% 1=identificator 2=number <sp>
\def\defBPfromSP#1#2{%
  \bgroup
  \dimen0=#2sp
  \dimen0=.013837\dimen0
  \dimen0=72\dimen0
  \expandafter\xdef\csname#1\endcsname{%
```

First published in *Zpravodaj* 17:2 (2007), pp. 67–72, as "Používáme pdfTeX V: aktuální pozice sazby"; translation by the author. Reprinted with permission.

[1] We use LaTeX packages or macros for commands not closely related to our topic: `eso-pic`, `ifpdf`, `afterpage` and `\InputIfFileExists`.

```
23     \expandafter\removePT\the\dimen0 }%
24   \egroup}
```

Now we are prepared to proceed to our topic — the current typesetting position. As we mentioned before, the position is not known until `\shipout` and thus it has to be saved to an auxiliary file. We name this file with the main file name and a `.pos` extension. The following macros open this file for writing at the document beginning and close it at the document end:

```
25 \newwrite\posHandle
26 \def\posFile{\jobname.pos }
27
28 \def\posOpen{\openout\posHandle=\posFile}
29 \def\posClose{\closeout\posHandle}
30
31 \AtBeginDocument{\posLoad\posOpen}
32 \AtEndDocument{\posClose}
```

The user macro `\posMark` writes the position to the file. It uses all three new pdfTEX primitives:

```
33 \def\posMark#1{% 1=place_label
34   \pdfsavepos
35   \write\posHandle{%
36     \string\posDef\string{#1\string}%
37     \string{\the\pdflastxpos\string}%
38     \string{\the\pdflastypos\string}}}
```

After the first LATEX run the following file is created:

```
39 \posDef{A}{10597449}{27447688}
40 \posDef{B}{24506216}{25133596}
```

This file is loaded by the `\posLoad` call at line 31:

```
41 \def\posLoad{\InputIfFileExists{\posFile}{}{}}
```

Loading the `.pos` file only if it exists avoids an error in the first LATEX pass.

The task of the internal `\posDef` macro, which is passed the label name and the (x, y) position, is to create two macros `\pos-x-sp-place_label` and `\pos-y-sp-place_label` with the values in *sp* and corresponding macros `\pos-x-bp-place_label` and `\pos-y-bp-place_label` in *bp*:

```
42 % 1=place_label 2=x-pos 3=y-pos
43 \def\posDef#1#2#3{%
44   \expandafter
45     \def\csname pos-x-sp-#1\endcsname{#2}%
46   \posDefXbp{#1}%
47   \expandafter
48     \def\csname pos-y-sp-#1\endcsname{#3}%
49   \posDefYbp{#1}}
```

Unit conversion is done by:

```
50 \def\posDefXbp#1% 1=place_label
51   {\defBPfromSP{pos-x-bp-#1}{\posGetX{#1}}}
52 \def\posDefYbp#1% 1=place_label
53   {\defBPfromSP{pos-y-bp-#1}{\posGetY{#1}}}
```

Macro calling is simplified by these definitions:

```
54 \def\posGetXY#1{\expandafter% 1=full_label
55   \ifx\csname #1\endcsname\relax0
56   \else\csname #1\endcsname\fi}
57 \def\posGetX#1{\posGetXY{pos-x-sp-#1}}
58 \def\posGetY#1{\posGetXY{pos-y-sp-#1}}
59 \def\posGetXbp#1{\posGetXY{pos-x-bp-#1}}
60 \def\posGetYbp#1{\posGetXY{pos-y-bp-#1}}
```

These previously defined absolute coordinates enter the macro for the line drawing. Its suitable placement is inside `\shipout`, when the base coordinate system is on. This is simplified by the package `eso-pic`:

```
61 % 1,2=start x y; 3,4=stop x y <bp unit>
62 \def\AbsLine#1,#2--#3,#4{%
63   \AddToShipoutPicture{%
64     \AtPageLowerLeft{\Line#1,#2--#3,#4}}}
```

And finally, we have the top-level drawing macro `\AbsLineFromTwoMarks` with the place labels as arguments:

```
65 % 1=place_label_A 2=place_label_B
66 \def\AbsLineFromTwoMarks#1#2{%
67   \AbsLine(\posGetXbp{#1},\posGetYbp{#1}--%
68            \posGetXbp{#2},\posGetYbp{#2})}
```

Now we can see that adding the next two lines after our illustrative paragraph will draw the line:

```
69 \AbsLineFromTwoMarks{A}{B}
70 \afterpage{\ClearShipoutPicture}
```

The last line avoids repeating the drawing on every subsequent page.

4 Conclusion

The current typesetting position is a useful pdfTEX extension. It works in both PDF output and DVI output from pdfTEX. Many graphical tricks such as framing word(s)/sentence(s)/paragraph(s) or surrounding them by backgrounds, emphasizing page parts by a vertical line in the margin, visualization of page elements relationship, and tabular cell placement can benefit from it. Here is a list of some LATEX packages that utilize this feature: `changebar`, `marginnote`, `t-angles`, `pdfsync`, `tabularht`. ConTEXt employs it throughout.

[*Editor's note:* This is one of a series of articles by Dr. Zyka on pdfTEX primitives. We hope to reprint other installments in future issues.]

◇ Vít Zýka
TYPOkvítek
Prague, Czech Republic
vit dot zyka (at) seznam dot cz