Displaying page layout variables

Kent McPherson a.o.[*]

1995/06/25

1 Introduction

This LaTeX 2ε package is a reimplementation of `layout.sty` by Kent McPherson. It defines the command `\layout` which produces an overview of the layout of the current document.

2 The implementation

This package prints a figure to illustrate the layout that is implemented by the document class. In the figure several words appear. They are stored in control sequences to be able to select a different language.

```
% \changes{v1.1f}{1995/03/14}{Added \cs{notshown}}
% \changes{v1.1f}{1995/03/14}{Added option french}
⟨*package⟩
\DeclareOption{dutch}{%
  \def\Headertext{Kopregel}
  \def\Bodytext{Broodtekst}
  \def\Footertext{Voetregel}
  \def\MarginNotestext{Marge\\Notities}
  \def\oneinchtext{een inch}
  \def\notshown{niet getoond}
  }
\DeclareOption{english}{%
  \def\Headertext{Header}
  \def\Bodytext{Body}
  \def\Footertext{Footer}
  \def\MarginNotestext{Margin\\Notes}
  \def\oneinchtext{one inch}
  \def\notshown{not shown}
  }
\DeclareOption{french}{%
  \def\Headertext{Ent\^{e}te}
  \def\Bodytext{Corps}
  \def\Footertext{Pied de page}
  \def\MarginNotestext{Marge\\Notes}
  \def\oneinchtext{un pouce}
  \def\notshown{non affich\'{e}}
```

[*] Converted for LaTeX 2ε by Johannes Braams

```
  }
\DeclareOption{francais}{\ExecuteOptions{french}}
```

This package has an option verbose. Using it will make the command `\layout` type some of the parameters on the terminal.

```
\DeclareOption{verbose}{\let\type\typeout}
\DeclareOption{silent}{\let\type\@gobble}
```

The normal behaviour of this package when showing the values of the parameters is to truncate them. However, if you want to see the real parameter values you can use the option reals to get that effect.

```
\def\lay@value{}
\DeclareOption{integers}{%
  \renewcommand*{\lay@value}[2]{%
    \expandafter\number\csname #1@#2\endcsname pt}}
\DeclareOption{reals}{%
  \renewcommand*{\lay@value}[2]{\the\csname #2\endcsname}}
```

The default language is English, the default mode is silent and the default way of showing parameter values is to use integers.

```
\ExecuteOptions{english,silent,integers}
\ProcessOptions
```

`\bs` Define `\bs` if it is undefined, redefine it if it is already defined.

```
\@ifundefined{bs}{\newcommand\bs{\char '134 }}%
  {\renewcommand\bs{\char '134 }}
```

`\ConvertToCount` This macro stores the value of a *length* register in a *count* register.

```
\def\ConvertToCount#1#2{%
```

First copy the value

```
  #1=#2
```

Then divide it by 65536.

```
  \divide #1 by 65536}
```

The result of this is that the *count* register holds the value of the *length* register in points.

`\SetToHalf` `\SetToQuart` Small macros used in computing positions.

```
\def\SetToHalf#1#2{#1=#2\relax\divide#1by\tw@}
\def\SetToQuart#1#2{#1=#2\relax\divide#1by4}
```

`\Identify` A small macro used in identifying dimensions.

```
\def\Identify#1{%
  \put(\PositionX,\PositionY){\circle{20}}
  \put(\PositionX,\PositionY){\makebox(0,0){\tiny #1}}
}
```

`\InsideHArrow` This macro is used to produce two horizontal arrows inside a box. The argument gives the width of the box.

```
\def\InsideHArrow#1{{%
  \ArrowLength = #1
  \divide\ArrowLength by \tw@
  \advance\ArrowLength by -10
```

```
  \advance\PositionX by -10
  \ifnum\ArrowLength<\z@
    \put(\PositionX,\PositionY){\vector(1,0){-\ArrowLength}}
    \advance\PositionX by 20
    \put(\PositionX,\PositionY){\vector(-1,0){-\ArrowLength}}
  \else
    \put(\PositionX,\PositionY){\vector(-1,0){\ArrowLength}}
    \advance\PositionX by 20
    \put(\PositionX,\PositionY){\vector(+1,0){\ArrowLength}}
  \fi
}}
```

`\InsideVArrow` This macro is used to produce two vertical arrows inside a box. The argument gives the height of the box.

```
\def\InsideVArrow#1{{%
  \ArrowLength = #1
  \divide\ArrowLength by \tw@
  \advance\ArrowLength by -10
  \advance\PositionY by -10
  \put(\PositionX,\PositionY){\vector(0,-1){\ArrowLength}}
  \advance\PositionY by 20
  \put(\PositionX,\PositionY){\vector(0,+1){\ArrowLength}}
}}
```

`\OutsideHArrow` This macro is used to produce two horizontal arrows to delimit a length. The first argument is the position for the right arrow, the second argument gives the length and the third specifies the length of the arrows.

```
\def\OutsideHArrow#1#2#3{{%
  \PositionX = #1
  \advance\PositionX by #3
  \put(\PositionX,\PositionY){\vector(-1,0){#3}}
  \PositionX = #1 \advance\PositionX-#2
  \advance\PositionX by -#3
  \put(\PositionX,\PositionY){\vector(+1,0){#3}}
}}
```

`\OutsideVArrow` This macro is used to produce two vertical arrows to delimit a length. The first argument is the position for the lower arrow, the second argument gives the length and the third and fourth specify the lenghts of the lower and upper arrow.

```
\def\OutsideVArrow#1#2#3#4{{%
  \PositionY = #1
  \advance\PositionY by -#3
  \put(\PositionX,\PositionY){\vector(0,+1){#3}}
  \PositionY = #1
  \advance\PositionY#2
  \advance\PositionY#4
  \put(\PositionX,\PositionY){\vector(0,-1){#4}}
}}
```

`\Show` Macro used in the table that shows the setting of the parameters.

```
\def\Show#1#2{\bs #2 = \lay@value{#1}{#2}}
```

`\Type` Macro used to show a setting of a parameter on the terminal.

```
\def\Type#1#2{%
  \type{#2 = \lay@value{#1}{#2}}}
```

`\oneinch` A constant, giving the length of an inch in points (approximately)

```
\newcount\oneinch
\oneinch=72
```

Because the overview of the layout is produced in a figure environment we need to allocate a number of counters that are used to store the values of various dimensions.

The dimensions of the paper

```
\newcount\cnt@paperwidth
\newcount\cnt@paperheight
\ConvertToCount\cnt@paperwidth\paperwidth
\ConvertToCount\cnt@paperheight\paperheight
```

the offsets,

```
\newcount\cnt@hoffset
\newcount\cnt@voffset
\ConvertToCount\cnt@hoffset\hoffset
\ConvertToCount\cnt@voffset\voffset
```

dimensions of the text area,

```
\newcount\cnt@textheight
\newcount\cnt@textwidth
\ConvertToCount\cnt@textheight\textheight
\ConvertToCount\cnt@textwidth\textwidth
```

margins,

```
\newcount\cnt@topmargin
\newcount\cnt@oddsidemargin
\newcount\cnt@evensidemargin
\ConvertToCount\cnt@topmargin\topmargin
\ConvertToCount\cnt@oddsidemargin\oddsidemargin
\ConvertToCount\cnt@evensidemargin\evensidemargin
```

dimensions of the running heads,

```
\newcount\cnt@headheight
\newcount\cnt@headsep
\ConvertToCount\cnt@headheight\headheight
\ConvertToCount\cnt@headsep\headsep
```

marginal paragraphs,

```
\newcount\cnt@marginparsep
\newcount\cnt@marginparwidth
\newcount\cnt@marginparpush
\ConvertToCount\cnt@marginparsep\marginparsep
\ConvertToCount\cnt@marginparwidth\marginparwidth
\ConvertToCount\cnt@marginparpush\marginparpush
```

the distance between the running footers and the text,

```
\newcount\cnt@footskip
\ConvertToCount\cnt@footskip\footskip
```

and the height of the footers, which is needed here to display a box, but which isn't used by LaTeX.

```
\newcount\fheight
\fheight=12
```

Apart from integer representations of the page layout parameters we also need registers to store reference values in.

The position of the top of the 'printable area' is one inch below the top of the paper by default. The value of `\ref@top` is relative to the lower left corner of the picture environment that will be used.

```
\newcount\ref@top
\ref@top=\cnt@paperheight \advance\ref@top by -\oneinch
```

For the offsets,

```
\newcount\ref@hoffset
\newcount\ref@voffset
```

The `\hoffset` and `\voffset` values are added to the default offset of one inch.

```
\ref@hoffset=\cnt@hoffset  \advance\cnt@hoffset by \oneinch
\ref@voffset=\cnt@voffset
```

`\cnt@voffset` is converted to be relative to the origin of the picture.

```
\cnt@voffset=\ref@top
\advance\cnt@voffset by -\ref@voffset
```

and the text areas, running heads,

```
\newcount\ref@head
\ref@head=\ref@top
  \advance\ref@head by -\ref@voffset
  \advance\ref@head by -\cnt@topmargin
  \advance\ref@head by -\cnt@headheight
```

body of the text

```
\newcount\ref@body
\ref@body=\ref@head
  \advance\ref@body by -\cnt@headsep
  \advance\ref@body by -\cnt@textheight
```

and running footers.

```
\newcount\ref@foot
  \ref@foot=\ref@body
  \advance\ref@foot by -\cnt@footskip
```

These are different for even and odd pages, so they are computed by `\layout`.

```
\newcount\ref@margin
\newcount\ref@marginwidth
\newcount\ref@marginpar
```

The following are a number of scratch registers, used in the positioning of the various pices of the picture..

```
\newcount\Interval
\newcount\ExtraYPos
\newcount\PositionX
\newcount\PositionY
\newcount\ArrowLength
```

`\layout` The command `\layout` makes the picture and table that display the current settings of the layout parameters.

```
\newcommand\layout{%
  \@layout
  \if@twoside
    \@layout
  \fi}
\newcommand\@layout{%
  \thispagestyle{empty}
```

The actions of `\layout` depend on the pagestyle.

```
  \if@twoside
    \ifodd\count\z@
```

Here we deal with an odd page in the twosided case.

```
      \typeout{Two-sided document style, odd page.}
```

So we compute `\ref@marginwidth`, `\ref@marginpar` and `\ref@margin`.

```
      \ref@marginwidth=\cnt@oddsidemargin
      \ref@marginpar=\oneinch
      \advance\ref@marginpar by \ref@hoffset
      \advance\ref@marginpar by \cnt@oddsidemargin
      \ref@margin\ref@marginpar
      \if@reversemargin
        \advance\ref@marginpar by -\cnt@marginparsep
        \advance\ref@marginpar by -\cnt@marginparwidth
      \else
        \advance\ref@marginpar by \cnt@textwidth
        \advance\ref@marginpar by \cnt@marginparsep
      \fi
    \else
```

Here we deal with an even page in the twosided case.

```
  \typeout{Two-sided document style, even page.}
```

So we compute `\ref@marginwidth`, `\ref@marginpar` and `\ref@margin`.

```
      \ref@marginwidth=\cnt@evensidemargin
      \ref@marginpar=\oneinch
      \advance\ref@marginpar by \ref@hoffset
      \advance\ref@marginpar by \cnt@evensidemargin
      \ref@margin\ref@marginpar
      \if@reversemargin
        \advance\ref@marginpar by \cnt@textwidth
        \advance\ref@marginpar by \cnt@marginparsep
      \else
        \advance\ref@marginpar by -\cnt@marginparsep
        \advance\ref@marginpar by -\cnt@marginparwidth
      \fi
    \fi
  \else
```

Finally we the case for single sided printing.

```
    \typeout{One-sided document style.}
    \ref@marginwidth=\cnt@oddsidemargin
    \ref@marginpar=\oneinch
    \advance\ref@marginpar by \ref@hoffset
```

```
    \advance\ref@marginpar by \cnt@oddsidemargin
    \ref@margin\ref@marginpar
    \if@reversemargin
      \advance\ref@marginpar by -\cnt@marginparsep
      \advance\ref@marginpar by -\cnt@marginparwidth
    \else
      \advance\ref@marginpar by \cnt@textwidth
      \advance\ref@marginpar by \cnt@marginparsep
    \fi
  \fi
```

Now we begin the picture environment; dividing all the lengths by two is done by setting `\unitlength` to `0.5pt`

```
  \setlength{\unitlength}{.5pt}
  \begin{picture}(\cnt@paperwidth,\cnt@paperheight)
    \centering
    \thicklines
```

First we have the pagebox and reference lines,

```
    \put(0,0){\framebox(\cnt@paperwidth,\cnt@paperheight){\mbox{}}}
    \put(0,\cnt@voffset){\dashbox{10}(\cnt@paperwidth,0){\mbox{}}}
    \put(\cnt@hoffset,0){\dashbox{10}(0,\cnt@paperheight){\mbox{}}}
```

then the header,

```
    \put(\ref@margin,\ref@head){%
      \framebox(\cnt@textwidth,\cnt@headheight)%
        {\footnotesize\Headertext}}
```

the body of the text area,

```
    \put(\ref@margin,\ref@body){%
      \framebox(\cnt@textwidth,\cnt@textheight){\Bodytext}}
```

the footer

```
    \put(\ref@margin,\ref@foot){%
      \framebox(\cnt@textwidth,\fheight){\footnotesize\Footertext}}
```

and the space for marginal notes.

```
    \put(\ref@marginpar,\ref@body){%
      \framebox(\cnt@marginparwidth,\cnt@textheight)%
               {\footnotesize\shortstack{\MarginNotestext}}}
```

Then we start putting in 'arrows' to mark the various parameters. From here we use `\thinlines`.

```
    \thinlines
```

`\PositionX` and `\PositionY` will be the coordinates of the center of the arrow displaying `\textwidth`.

```
    \SetToHalf\PositionX\cnt@textwidth
    \advance\PositionX by \ref@margin
```

The arrow should be a bit above the bottom of the 'body box'.

```
    \PositionY = \ref@body
    \advance\PositionY by 50
```

An identifying number is put here, in a circle.

```
    \Identify{8}
```

Then the arrow is drawn.

```
  \InsideHArrow\cnt@textwidth
```

Now the `\textheight`

```
  \SetToHalf\PositionY\cnt@textheight
  \advance\PositionY by \ref@body
```

The x-psition of the arrow is at 4/5 of the width of the 'body box'.

```
  \PositionX = \cnt@textwidth
  \divide\PositionX by 5
  \multiply \PositionX by 4
  \advance\PositionX by \ref@margin
```

An identifying number is put here, in a circle.

```
  \Identify{7}
  \InsideVArrow\cnt@textheight
```

The `\hoffset`,

```
  \PositionY = \ref@foot
  \SetToHalf\PositionX\cnt@hoffset
  \Identify{1}
  \InsideHArrow\cnt@hoffset
```

The width of the margin.

```
  \SetToQuart\PositionY\cnt@textheight
  \advance\PositionY by \ref@body
  \OutsideHArrow\ref@margin\ref@marginwidth{20}
  \PositionX = \cnt@hoffset
  \advance\PositionX by -30
  \Identify{3}
```

the `\marginparwidth`,

```
  \SetToQuart\PositionY\cnt@textheight
  \advance\PositionY by \ref@body
```

This arrow has to be bit below the one for the `\oddsidemargin` or `\evensidemargin`.

```
  \advance\PositionY by 30
  \SetToHalf\PositionX\cnt@marginparwidth
  \advance\PositionX by \ref@marginpar
  \Identify{10}
  \InsideHArrow\cnt@marginparwidth
```

The `\marginparsep`, this depends on single or double sided printing.

```
  \advance\PositionY by 30
  \if@twoside
    \ifodd\count\z@
```

Twosided mode, odd page;

```
      \OutsideHArrow\ref@marginpar\cnt@marginparsep{20}
      \PositionX = \ref@marginpar
    \else
```

and even page.

```
      \OutsideHArrow\ref@margin\cnt@marginparsep{20}
      \PositionX = \ref@margin
    \fi
  \else
```

Single sided mode.

```
    \OutsideHArrow\ref@marginpar\cnt@marginparsep{20}
    \PositionX = \ref@marginpar
  \fi
  \advance\PositionX by 30
  \Identify{9}
```

Identify the `\footskip`. The arrow will be located on 1/8th of the `\textwidth`.

```
  \PositionX = \cnt@textwidth
  \divide\PositionX by 8
  \advance\PositionX by \ref@margin
  \OutsideVArrow\ref@foot\cnt@footskip{20}{20}
  \PositionY = \ref@foot
  \advance\PositionY by -30
  \Identify{11}
```

Identify the `\voffset`. The arrow will be located a bit to the left of the edge of the paper.

```
  \PositionX = \cnt@paperwidth
  \advance\PositionX by -50
  \PositionY = \cnt@paperheight
  \ExtraYPos = \PositionY
  \advance\ExtraYPos by -\cnt@voffset
  \advance\PositionY by \cnt@voffset
  \divide\PositionY by \tw@
  \Identify{2}
  \InsideVArrow\ExtraYPos
```

Identify `\topmargin`, `\headheight` and `\headsep`.

The arrows will be located on 1/8th of the `\textwidth`, with intervals of the same size, stored in `\Interval`.

```
  \Interval = \cnt@textwidth
  \divide\Interval by 8
  \PositionX = \ref@margin
  \advance\PositionX by \Interval
```

First the `\topmargin`. The length of upper arrows for these three parameters dimensions are different. In the computation of these lengths we have to compensate for the diameter of the cirle.

```
  \Identify{4}
  \ExtraYPos = \ref@head
  \advance\ExtraYPos\cnt@headheight
  \ArrowLength = \PositionY
  \advance\ArrowLength-\ExtraYPos
  \advance\ArrowLength-\cnt@topmargin
  \advance\ArrowLength-10%
  \OutsideVArrow\ExtraYPos\cnt@topmargin{20}{\ArrowLength}
  \advance\PositionX by \Interval
```

Then the `\headheight`

```
  \Identify{5}
  \advance\ArrowLength\cnt@topmargin
  \OutsideVArrow\ref@head\cnt@headheight{20}{\ArrowLength}
  \advance\PositionX by \Interval
```

and finally the `\headsep`

```
    \Identify{6}
    \advance\ArrowLength\cnt@headheight
    \ExtraYPos=\ref@body
    \advance\ExtraYPos\cnt@textheight
    \OutsideVArrow\ExtraYPos\cnt@headsep{20}{\ArrowLength}
```

Here we can end the picture environment and insert a little space.

```
  \end{picture}

  \medskip
```

Below the picture we put a table to show the actual values of the parameters. Note that fractional points are truncated, i.e., `72.27pt` is displayed as `72pt`

The table is typeset inside a box with a depth of 0 to always keep it on the same page as the picture.

```
  \vtop to 0pt{%
    \@minipagerestore\footnotesize\ttfamily
    \begin{tabular}{@{}rl@{\hspace{20pt}}rl}
      1 & \oneinchtext\ + \bs\texttt{hoffset}
        & 2 & \oneinchtext\ + \bs\texttt{voffset} \\
      3 & \if@twoside
            \ifodd\count\z@ \Show{cnt}{oddsidemargin}
            \else \Show{cnt}{evensidemargin}
            \fi
          \else
            \Show{cnt}{oddsidemargin}
          \fi                    & 4 & \Show{cnt}{topmargin} \\
      5 & \Show{cnt}{headheight} & 6 & \Show{cnt}{headsep} \\
      7 & \Show{cnt}{textheight} & 8 & \Show{cnt}{textwidth} \\
      9 & \Show{cnt}{marginparsep}&10& \Show{cnt}{marginparwidth} \\
      11& \Show{cnt}{footskip}   &   & \Show{cnt}{marginparpush}
       \rlap{(\notshown)}\\
        & \Show{ref}{hoffset}    &   & \Show{ref}{voffset} \\
        & \Show{cnt}{paperwidth} &   & \Show{cnt}{paperheight} \\

  \end{tabular}\vss}
```

When the option verbose was used the following lines will show dimensions on the terminal.

```
  \Type{ref}{hoffset}
  \Type{ref}{voffset}
  \Type{cnt}{textheight}
  \Type{cnt}{textwidth}
```

Finally we start a new page.

```
  \newpage
}
⟨/package⟩
```