The geometry package

Hideo Umeki
https://github.com/davidcarlisle/geometry

2020/01/02 v5.9

Abstract

This package provides a flexible and easy interface to page dimensions. You can change the page layout with intuitive parameters. For instance, if you want to set a margin to 2cm from each edge of the paper, you can type just `\usepackage[margin=2cm]{geometry}`. The page layout can be changed in the middle of the document with `\newgeometry` command.

1 Preface to version 5

- **Changing page layout mid-document.**

  The new commands `\newgeometry{···}` and `\restoregeometry` allow users to change page dimensions in the middle of the document. `\newgeometry` is almost similar to `\geometry` except that `\newgeometry` disables all the options specified in the preamble and skips the papersize-related options: `landscape`, `portrait` and paper size options (such as `papersize`, `paper=a4paper` and so forth).

- **A new set of options to specify the layout area.**

  The options specified for the area, in which the page dimensions are calculated, are added: layout, layoutsize, layoutwidth, layoutheight and so forth. These options would help to print the specified layout to a different sized paper. For example, with `a4paper` and `layout=a5paper`, the geometry package uses 'A5' layout to calculate margins with the paper size still 'A4'.

- **A new driver option `xetex`.**

  The new driver option `xetex` is added. The driver auto-detection routine has been revised so as to avoid an error with undefined control sequences. Note that 'geometry.cfg' in TEX Live, which disables the auto-detection routine and sets `pdftex`, is no longer necessary and has no problem even though it still exists. To set `xetex` is strongly recommended with XeLaTeX.

- **New paper size presets for JIS B-series and ISO C-series.**

  The papersize presets `b0j` to `b6j` for JIS (Japanese Industrial Standards) B-series and `c0paper` to `c6paper` for ISO C-series (v5.4∼) are added.

- **Changing default for underspecified margin.**

  In the previous version, if only one margin was specified, `bottom=1cm` for example, then geometry set the other margin with the margin ratio (1:1 by default for the vertical dimensions) and got `top=1cm` in this case. The version 5 sets the text-body size with the default `scale` (= 0.7) and determine the unspecified margin. (See Section 6.5)

- **The option `showframe` and `showcrop` works on every page.**

  With `showframe` option, the page frames are shown on every page. In addition, a new option `showcrop` prints crop marks at each corner of layout area on every page. Note that the marks would be invisible without specifying the layout size smaller than paper size. Version 5.4 introduced a new `\shipout` overloading process using atbegshi package, so the atbegshi package is required when showframe or showcrop option is specified.

- **Loading geometry.cfg precedes processing class options.**

  The previous version loaded geometry.cfg after processing the document class options. Now that the config file is loaded before processing the class options, you can change the behavior specified in geometry.cfg by adding options into `\documentclass` as well as `\usepackage` and `\geometry`.

- **Deleted options: `compat2` and `twosideshift`.** The version 5 has no longer compatibility with the previous ones. `compat2` and `twosideshift` are gone for simplicity.

2 Introduction

To set dimensions for page layout in LaTeX is not straightforward. You need to adjust several LaTeX native dimensions to place a text area where you want. If you want to center the text area in the paper you use, for example, you have to specify native dimensions as follows:

```
\usepackage{calc}
\setlength\textwidth{7in}
\setlength\textheight{10in}
\setlength\oddsidemargin{(\paperwidth-\textwidth)/2 - 1in}
\setlength\topmargin{(\paperheight-\textheight
                      -\headheight-\headsep-\footskip)/2 - 1in}.
```

Without package *calc*, the above example would need more tedious settings. Package geometry provides an easy way to set page layout parameters. In this case, what you have to do is just

```
\usepackage[text={7in,10in},centering]{geometry}.
```

Besides centering problem, setting margins from each edge of the paper is also troublesome. But geometry also make it easy. If you want to set each margin to 1.5in, you can type

```
\usepackage[margin=1.5in]{geometry}
```

Thus, the geometry package has an auto-completion mechanism, in which unspecified dimensions are automatically determined. The geometry package will be also useful when you have to set page layout obeying the following strict instructions: for example,

> *The total allowable width of the text area is* 6.5 *inches wide by* 8.75 *inches high. The top margin on each page should be* 1.2 *inches from the top edge of the page. The left margin should be* 0.9 *inch from the left edge. The footer with page number should be at the bottom of the text area.*

In this case, using geometry you can type

```
\usepackage[total={6.5in,8.75in},
            top=1.2in, left=0.9in, includefoot]{geometry}.
```

Setting a text area on the paper in document preparation system has some analogy to placing a window on the background in the window system. The name 'geometry' comes from the `-geometry` option used for specifying a size and location of a window in X Window System.

3 Page geometry

Figure 1 shows the page layout dimensions defined in the geometry package. The page layout contains a *total body* (printable area) and *margins*. The *total body* consists of a *body* (text area) with an optional *header*, *footer* and marginal notes (marginpar). There are four margins: *left*, *right*, *top* and *bottom*. For twosided documents, horizontal margins should be called *inner* and *outer*.

paper	:	*total body* and *margins*
total body	:	*body* (text area) (optional *head*, *foot* and *marginpar*)
margins	:	*left* (*inner*), *right* (*outer*), *top* and *bottom*

Each margin is measured from the corresponding edge of a paper. For example, left margin (inner margin) means a horizontal distance between the left (inner) edge of the paper and that of the total body. Therefore the left and top margins defined in geometry are different from the native dimensions `\leftmargin` and `\topmargin`. The size of a body (text area) can be modified by `\textwidth` and `\textheight`. The dimensions for paper, total body and margins have the following relations.

$$\texttt{paperwidth} = \texttt{left} + \texttt{width} + \texttt{right} \quad (1)$$

$$\texttt{paperheight} = \texttt{top} + \texttt{height} + \texttt{bottom} \quad (2)$$

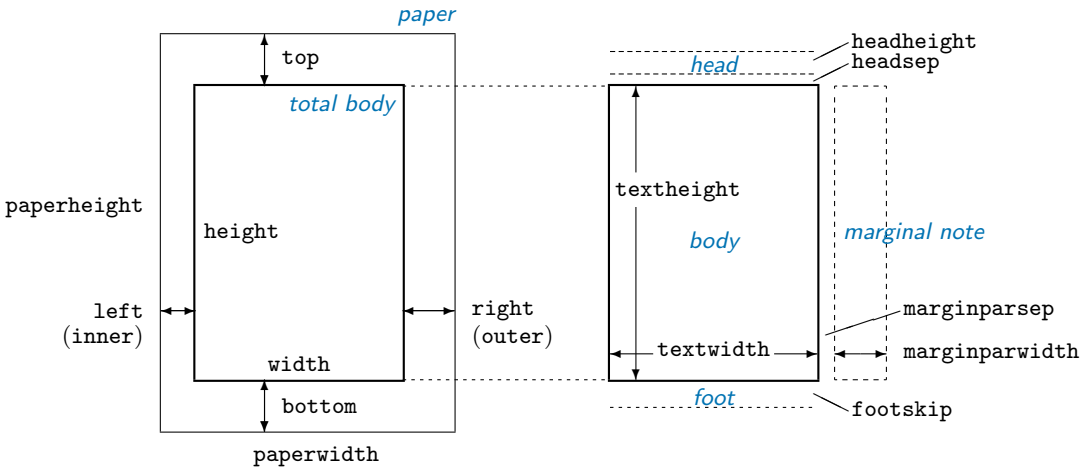


Figure 1: Dimension names used in the geometry package. `width` = `textwidth` and `height` = `textheight` by default. `left`, `right`, `top` and `bottom` are margins. If margins on verso pages are swapped by `twoside` option, margins specified by `left` and `right` options are used for the inside and outside margins respectively. `inner` and `outer` are aliases of `left` and `right` respectively.

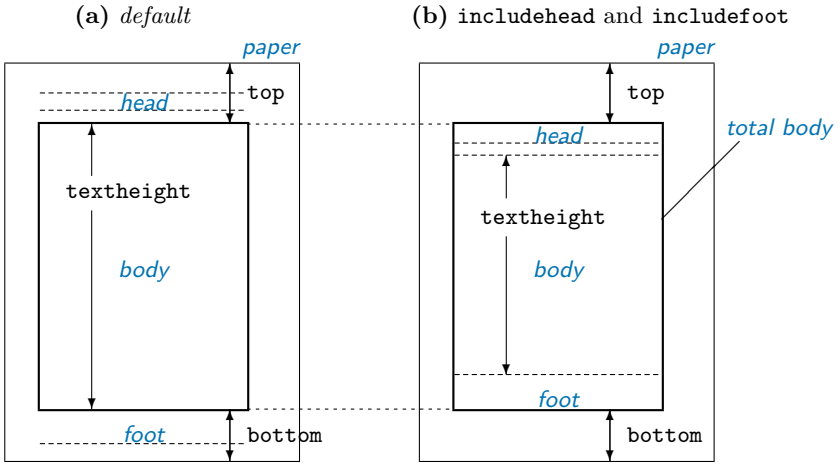


Figure 2: `includehead` and `includefoot` include the head and foot respectively into *total body*. **(a)** `height` = `textheight` (default). **(b)** `height` = `textheight` + `headheight` + `headsep` + `footskip` if `includehead` and `includefoot`. If the top and bottom margins are specified, `includehead` and `includefoot` result in shorter `textheight`.

The total body `width` and `height` would be defined:

$$\texttt{width} := \texttt{textwidth} \quad (+\texttt{marginparsep} + \texttt{marginparwidth}) \tag{3}$$

$$\texttt{height} := \texttt{textheight} \quad (+\texttt{headheight} + \texttt{headsep} + \texttt{footskip}) \tag{4}$$

In Equation (3) `width:=textwidth` by default, while `marginparsep` and `marginparwidth` are included in `width` if `includemp` option is set `true`. In Equation (4), `height:=textheight` by default. If `includehead` is set to `true`, `headheight` and `headsep` are considered as a part of `height`. In the same way, `includefoot` takes `footskip` into `height`. Figure 2 shows how these options work in the vertical direction.

Thus, the page layout consists of three parts (lengths) in each direction: one body and two margins. If the two of them are explicitly specified, the other length is obvious and no need to be specified. Figure 3 shows a simple model of page dimensions. When a length `L` is given and is partitioned into the body `b`, the margins `a` and `c`, it's obvious that

$$\mathtt{L} = \mathtt{a} + \mathtt{b} + \mathtt{c} \tag{5}$$

The specification with two of the three (`a`,`b` and `c`) fixed explicitly is solvable. If two or more are left unspecified or 'underspecified', Equation (5) cannot be solved without any other relation between them. If all of them are specified, then it needs to check whether or not they satisfy Equation (5), that is too much specification or 'overspecified'.

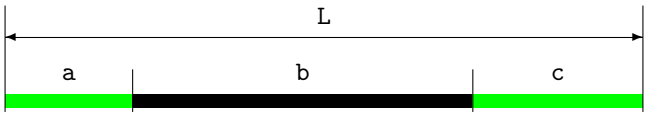


Figure 3: A simple model of page dimensions.

The geometry package has auto-completion mechanism that saves the trouble of specifying the page layout dimensions. For example, you can set

```
\usepackage[width=14cm, left=3cm]{geometry}
```

on A4 paper. In this case you don't have to set the right margin The details of auto-completion will be described in Section 6.5.

4 User interface

4.1 Commands

The geometry package provides the following commands:

- `\geometry{`⟨*options*⟩`}`
- `\newgeometry{`⟨*options*⟩`}` and `\restoregeometry`
- `\savegeometry{`⟨*name*⟩`}` and `\loadgeometry{`⟨*name*⟩`}`

`\geometry{`⟨*options*⟩`}` changes the page layout according to the options specified in the argument. This command, if any, should be placed only in the preamble (before `\begin{document}`).

The geometry package may be used as part of a class or another package you use in your document. The command `\geometry` can overwrite some of the settings in the preamble. Multiple use of `\geometry` is allowed and then processed with the options concatenated. If geometry is not yet loaded, you can use only `\usepackage[`⟨*options*⟩`]{geometry}` instead of `\geometry`.

`\newgeometry{`⟨*options*⟩`}` changes the page layout mid-document. `\newgeometry` is almost similar to `\geometry` except that `\newgeometry` disables all the options specified by `\usepackage` and `\geometry` in the preamble and skips papersize-related options. `\restoregeometry` restores the page layout specified in the preamble. This command has no arguments. See Section 7 for details.

`\savegeometry{`⟨*name*⟩`}` saves the page dimensions as ⟨*name*⟩ where you put this command. `\loadgeometry{`⟨*name*⟩`}` loads the page dimensions saved as ⟨*name*⟩. See Section 7 for details.

4.2 Optional argument

The geometry package adopts keyval interface '⟨*key*⟩=⟨*value*⟩' for the optional argument to `\usepackage`, `\geometry` and `\newgeometry`.

The argument includes a list of comma-separated keyval options and has basic rules as follows:

- Multiple lines are allowed, while blank lines are not.
- Any spaces between words are ignored.
- Options are basically order-independent. (There are some exceptions. See Section 6.2 for details.)

For example,

```
\usepackage[ a5paper ,  hmargin = { 3cm,
                .8in } , height
         =  10in ]{geometry}
```

is equivalent to

```
\usepackage[height=10in,a5paper,hmargin={3cm,0.8in}]{geometry}
```

Some options are allowed to have sub-list, e.g. `{3cm,0.8in}`. Note that the order of values in the sub-list is significant. The above setting is also equivalent to the followings:

```
\usepackage{geometry}
\geometry{height=10in,a5paper,hmargin={3cm,0.8in}}
```

or

```
\usepackage[a5paper]{geometry}
\geometry{hmargin={3cm,0.8in},height=8in}
\geometry{height=10in}.
```

Thus, multiple use of `\geometry` just appends options.

geometry supports package *calc*[1]. For example,

```
\usepackage{calc}
\usepackage[textheight=20\baselineskip+10pt]{geometry}
```

4.3 Option types

geometry options are categorized into four types:

1. **Boolean type**

   takes a boolean value (`true` or `false`). If no value, `true` is set by default.

   ⟨*key*⟩`=true` | `false`.
   ⟨*key*⟩ with no value is equivalent to ⟨*key*⟩`=true`.

   Examples: `verbose=true`, `includehead`, `twoside=false`.
   Paper name is the exception. The preferred paper name should be set with no values. Whatever value is given, it is ignored. For instance, `a4paper=XXX` is equivalent to `a4paper`.

2. **Single-valued type**

   takes a mandatory value.

   ⟨*key*⟩`=`⟨*value*⟩.

   Examples: `width=7in`, `left=1.25in`, `footskip=1cm`, `height=.86\paperheight`.

3. **Double-valued type**

   takes a pair of comma-separated values in braces. The two values can be shortened to one value if they are identical.

   ⟨*key*⟩`={`⟨*value1*⟩`,`⟨*value2*⟩`}`.
   ⟨*key*⟩`=`⟨*value*⟩ is equivalent to ⟨*key*⟩`={`⟨*value*⟩`,`⟨*value*⟩`}`.

   Examples: `hmargin={1.5in,1in}`, `scale=0.8`, `body={7in,10in}`.

4. **Triple-valued type**

   takes three mandatory, comma-separated values in braces.

   ⟨*key*⟩`={`⟨*value1*⟩`,`⟨*value2*⟩`,`⟨*value3*⟩`}`

   Each value must be a dimension or null. When you give an empty value or '`*`', it means null and leaves the appropriate value to the auto-completion mechanism. You need to specify at least one dimension, typically two dimensions. You can set nulls for all the values, but it makes no sense.
   Examples:
   `hdivide={2cm,*,1cm}`, `vdivide={3cm,19cm, }`, `divide={1in,*,1in}`.

5 Option details

This section describes all options available in geometry. Options with a dagger † are not available as arguments of `\newgeometry` (See Section 7).

[1] CTAN: `macros/latex/required/tools`

5.1 Paper size

The options below set paper/media size and orientation.

† `paper` | `papername`
: specifies the paper size by name. `paper=`⟨*paper-name*⟩. For convenience, you can specify the paper name without `paper=`. For example, `a4paper` is equivalent to `paper=a4paper`.

† `a0paper, a1paper, a2paper, a3paper, a4paper, a5paper, a6paper,`
`b0paper, b1paper, b2paper, b3paper, b4paper, b5paper, b6paper,`
`c0paper, c1paper, c2paper, c3paper, c4paper, c5paper, c6paper,`
`b0j, b1j, b2j, b3j, b4j, b5j, b6j,`
`ansiapaper, ansibpaper, ansicpaper, ansidpaper, ansiepaper,`
`letterpaper, executivepaper, legalpaper`
: specifies paper name. The value part is ignored even if any. For example, the followings have the same effect: `a5paper`, `a5paper=true`, `a5paper=false` and so forth. `a[0-6]paper`, `b[0-6]paper` and `c[0-6]paper` are ISO A, B and C series of paper sizes respectively. The JIS (Japanese Industrial Standards) A-series is identical to the ISO A-series, but the JIS B-series is different from the ISO B-series. `b[0-6]j` should be used for the JIS B-series.

† `screen`
: a special paper size with (W,H) = (225mm,180mm). For presentation with PC and video projector, "`screen,centering`" with 'slide' documentclass would be useful.

† `paperwidth`
: width of the paper. `paperwidth=`⟨*length*⟩.

† `paperheight`
: height of the paper. `paperheight=`⟨*length*⟩.

† `papersize`
: width and height of the paper. `papersize={`⟨*width*⟩`,`⟨*height*⟩`}` or `papersize=`⟨*length*⟩.

† `landscape`
: switches the paper orientation to landscape mode.

† `portrait`
: switches the paper orientation to portrait mode. This is equivalent to `landscape=false`.

The options for paper names (e.g., `a4paper`) and orientation (`portrait` and `landscape`) can be set as document class options. For example, you can set `\documentclass[a4paper,landscape]{article}`, then `a4paper` and `landscape` are processed in `geometry` as well. This is also the case for `twoside` and `twocolumn` (see also Section 5.5).

5.2 Layout size

You can specify the layout area with options described in this section regardless of the paper size. The options would help to print the specified layout to a different sized paper. For example, with `a4paper` and `layout=a5paper`, the package uses 'A5' layout to calculate margins on 'A4' paper. The layout size defaults to the same as the paper. The options for the layout size are available in `\newgeometry`, so that you can change the layout size in the middle of the document. The paper size itself can't be changed though. Figure 4 shows what the difference between `layout` and `paper` is.

`layout`
: specifies the layout size by paper name. `layout=`⟨*paper-name*⟩. All the paper names defined in `geometry` are available. See Section 5.1 for details.

`layoutwidth`
: width of the layout. `layoutwidth=`⟨*length*⟩.

`layoutheight`
: height of the layout. `layoutheight=`⟨*length*⟩.

`layoutsize`
: width and height of the layout. `layoutsize={`⟨*width*⟩`,`⟨*height*⟩`}` or `layoutsize=`⟨*length*⟩.

`layouthoffset`
: specifies the horizontal offset from the left edge of the paper. `layouthoffset=`⟨*length*⟩.

`layoutvoffset`
: specifies the vertical offset from the top edge of the paper. `layoutvoffset=`⟨*length*⟩.

`layoutoffset`
: specifies both horizontal and vertical offsets. `layoutoffset={`⟨*hoffset*⟩`,`⟨*voffset*⟩`}` or `layoutsize=`⟨*length*⟩.

5.3 Body size

The options specifying the size of *total body* are described in this section.

`hscale`
: ratio of width of *total body* to `\paperwidth`. `hscale=`⟨*h-scale*⟩, e.g., `hscale=0.8` is equivalent to `width=0.8\paperwidth`. (`0.7` by default)

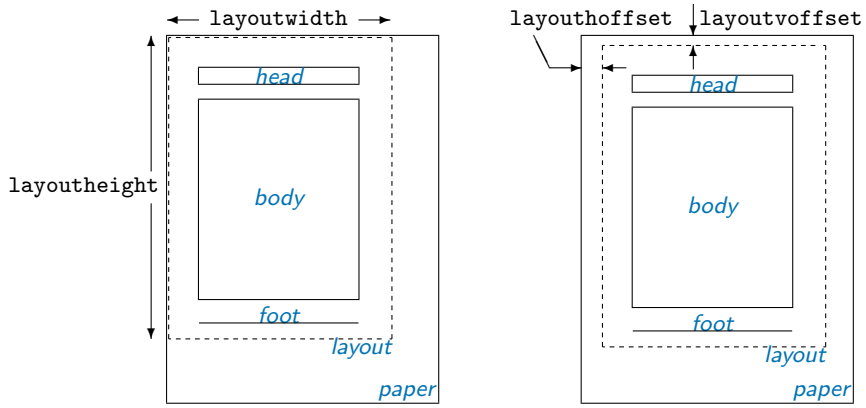


Figure 4: The dimensions related to the layout size. Note that the layout size defaults to the same size as the paper, so you don't have to specify layout-related options explicitly in most cases.

`vscale` ratio of height of *total body* to `\paperheight`, e.g., `vscale=`⟨*v-scale*⟩. (`0.7` by default) `vscale=0.9` is equivalent to `height=0.9\paperheight`.

`scale` ratio of *total body* to the paper. `scale={`⟨*h-scale*⟩`,`⟨*v-scale*⟩`}` or `scale=`⟨*scale*⟩. (`0.7` by default)

`width` | `totalwidth`
width of *total body*. `width=`⟨*length*⟩ or `totalwidth=`⟨*length*⟩. This dimension defaults to `textwidth`, but if `includemp` is set to `true`, `width` ≥ `textwidth` because `width` includes the width of the marginal notes. If `textwidth` and `width` are specified at the same time, `textwidth` takes priority over `width`.

`height` | `totalheight`
height of *total body*, excluding header and footer by default. If `includehead` or `includefoot` is set, `height` includes the head or foot of the page as well as `textheight`. `height=`⟨*length*⟩ or `totalheight=`⟨*length*⟩. If both `textheight` and `height` are specified, `height` will be ignored.

`total` width and height of *total body*. `total={`⟨*width*⟩`,`⟨*height*⟩`}` or `total=`⟨*length*⟩.

`textwidth` specifies `\textwidth`, the width of *body* (the text area). `textwidth=`⟨*length*⟩.

`textheight` specifies `\textheight`, the height of *body* (the text area). `textheight=`⟨*length*⟩.

`text` | `body` specifies both `\textwidth` and `\textheight` of the body of page. `body={`⟨*width*⟩`,`⟨*height*⟩`}` or `text=`⟨*length*⟩.

`lines` enables users to specify `\textheight` by the number of lines. `lines=`⟨*integer*⟩.

`includehead` includes the head of the page, `\headheight` and `\headsep`, into *total body*. It is set to `false` by default. It is opposite to `ignorehead`. See Figure 2 and Figure 5.

`includefoot` includes the foot of the page, `\footskip`, into *total body*. It is opposite to `ignorefoot`. It is `false` by default. See Figure 2 and Figure 5.

`includeheadfoot`
sets both `includehead` and `includefoot` to `true`, which is opposite to `ignoreheadfoot`. See Figure 2 and Figure 5.

`includemp` includes the margin notes, `\marginparwidth` and `\marginparsep`, into *body* when calculating horizontal calculation.

`includeall` sets both `includeheadfoot` and `includemp` to `true`. See Figure 5.

`ignorehead` disregards the head of the page, `headheight` and `headsep`, in determining vertical layout, but does not change those lengths. It is equivalent to `includehead=false`. It is set to `true` by default. See also `includehead`.

`ignorefoot` disregards the foot of page, `footskip`, in determining vertical layout, but does not change that length. This option defaults to `true`. See also `includefoot`.

`ignoreheadfoot`
sets both `ignorehead` and `ignorefoot` to `true`. See also `includeheadfoot`.

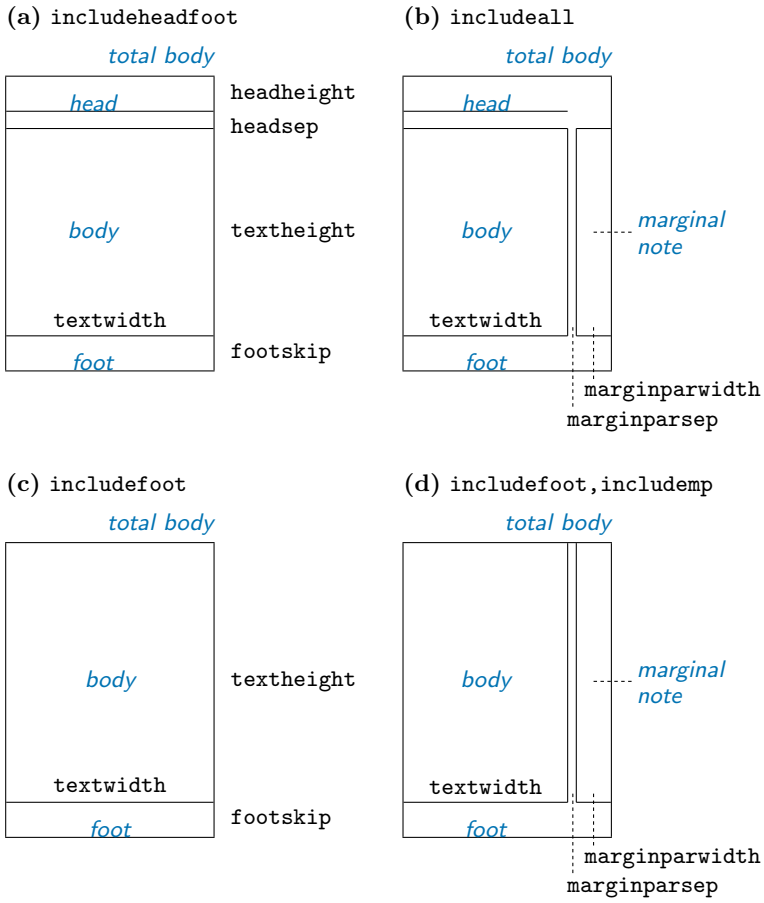


Figure 5: Sample layouts for *total body* with different switches. (a) `includeheadfoot`, (b) `includeall`, (c) `includefoot` and (d) `includefoot,includemp`. If `reversemp` is set to `true`, the location of the marginal notes are swapped on every page. Option `twoside` swaps both margins and marginal notes on verso pages. Note that the marginal note, if any, is printed despite `ignoremp` or `includemp=false` and overrun the page in some cases.

`ignoremp` disregards the marginal notes in determining the horizontal margins (defaults to `true`). If marginal notes overrun the page, the warning message will be displayed when `verbose=true`. See also `includemp` and Figure 5.

`ignoreall` sets both `ignoreheadfoot` and `ignoremp` to `true`. See also `includeall`.

`heightrounded`
This option rounds `\textheight` to n-times (n: an integer) of `\baselineskip` plus `\topskip` to avoid "underfull vbox" in some cases. For example, if `\textheight` is 486pt with `\baselineskip` 12pt and `\topskip` 10pt, then

$$(39 \times 12\text{pt} + 10\text{pt} =)\ 478\text{pt} < 486\text{pt} < 490\text{pt}\ (= 40 \times 12\text{pt} + 10\text{pt}),$$

as a result `\textheight` is rounded to 490pt. `heightrounded=false` by default.

Figure 5 illustrates various layouts with different layout modes. The dimensions for a header and a footer can be controlled by `nohead` or `nofoot` mode, which sets each length to 0pt directly. On the other hand, options with the prefix `ignore` do *not* change the corresponding native dimensions.

The following options can specify body and margins simultaneously with three comma-separated values in braces.

`hdivide` horizontal partitions (left,width,right). `hdivide={`⟨*left margin*⟩`,`⟨*width*⟩`,`⟨*right margin*⟩`}`. Note that you should not specify all of the three parameters. The best way of using this option is to specify two of three and leave the rest with null(nothing) or '`*`'. For example, when you set `hdivide={2cm,15cm, }`, the margin from the right-side edge of page will be determined calculating `paperwidth-2cm-15cm`.

`vdivide` vertical partitions (top,height,bottom). `vdivide={`⟨*top margin*⟩`,`⟨*height*⟩`,`⟨*bottom margin*⟩`}`.

`divide` `divide={`*A*`,`*B*`,`*C*`}` is interpreted as `hdivide={`*A*`,`*B*`,`*C*`}` and `vdivide={`*A*`,`*B*`,`*C*`}`.

5.4 Margin size

The options specifying the size of the margins are listed below.

`left` | `lmargin` | `inner`
left margin (for oneside) or inner margin (for twoside) of *total body*. In other words, the distance between the left (inner) edge of the paper and that of *total body*. `left=`⟨*length*⟩. `inner` has no special meaning, just an alias of `left` and `lmargin`.

`right` | `rmargin` | `outer`
right or outer margin of *total body*. `right=`⟨*length*⟩.

`top` | `tmargin` top margin of the page. `top=`⟨*length*⟩. Note this option has nothing to do with the native dimension `\topmargin`.

`bottom` | `bmargin`
bottom margin of the page. `bottom=`⟨*length*⟩.

`hmargin` left and right margin. `hmargin={`⟨*left margin*⟩`,`⟨*right margin*⟩`}` or `hmargin=`⟨*length*⟩.

`vmargin` top and bottom margin. `vmargin={`⟨*top margin*⟩`,`⟨*bottom margin*⟩`}` or `vmargin=`⟨*length*⟩.

`margin` `margin={`*A*`,`*B*`}` is equivalent to `hmargin={`*A*`,`*B*`}` and `vmargin={`*A*`,`*B*`}`. `margin=`*A* is automatically expanded to `hmargin=`*A* and `vmargin=`*A*.

`hmarginratio` horizontal margin ratio of `left` (inner) to `right` (outer). The value of ⟨*ratio*⟩ should be specified with colon-separated two values. Each value should be a positive integer less than 100 to prevent arithmetic overflow, e.g., `2:3` instead of `1:1.5`. The default ratio is `1:1` for oneside, `2:3` for twoside.

`vmarginratio` vertical margin ratio of `top` to `bottom`. The default ratio is `2:3`.

`marginratio` | `ratio`
horizontal and vertical margin ratios. `marginratio={`⟨*horizontal ratio*⟩`,`⟨*vertical ratio*⟩`}` or `marginratio=`⟨*ratio*⟩.

`hcentering` sets auto-centering horizontally and is equivalent to `hmarginratio=1:1`. It is set to `true` by default for oneside. See also `hmarginratio`.

`vcentering` sets auto-centering vertically and is equivalent to `vmarginratio=1:1`. The default is `false`. See also `vmarginratio`.

`centering` sets auto-centering and is equivalent to `marginratio=1:1`. See also `marginratio`. The default is `false`. See also `marginratio`.

`twoside` switches on twoside mode with left and right margins swapped on verso pages. The option sets `\@twoside` and `\@mparswitch` switches. See also `asymmetric`.

`asymmetric` implements a twosided layout in which margins are not swapped on alternate pages (by setting `\oddsidemargin` to `\evensidemargin + bindingoffset`) and in which the marginal notes stay always on the same side. This option can be used as an alternative to the twoside option. See also `twoside`.

`bindingoffset`
removes a specified space from the lefthand-side of the page for oneside or the inner-side for twoside. `bindingoffset=`⟨*length*⟩. This is useful if pages are bound by a press binding (glued, stitched, stapled ...). See Figure 6.

`hdivide` See description in Section 5.3.

`vdivide` See description in Section 5.3.

`divide` See description in Section 5.3.

5.5 Native dimensions

The options below overwrite LaTeX native dimensions and switches for page layout (See the right-hand side in Figure 1).

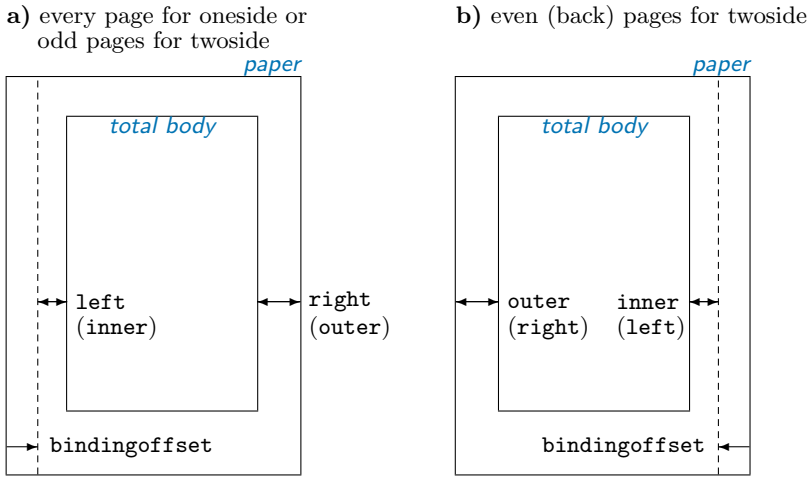


Figure 6: The option `bindingoffset` adds the specified length to the inner margin. Note that `twoside` option swaps the horizontal margins and the marginal notes together with `bindingoffset` on even pages (see **b)**), but `asymmetric` option suppresses the swap of the margins and marginal notes (but `bindingoffset` is still swapped).

`headheight` | `head`
: modifies `\headheight`, height of header. `headheight=`⟨*length*⟩ or `head=`⟨*length*⟩.

`headsep`
: modifies `\headsep`, separation between header and text (body). `headsep=`⟨*length*⟩.

`footskip` | `foot`
: modifies `\footskip`, distance separation between baseline of last line of text and baseline of footer. `footskip=`⟨*length*⟩ or `foot=`⟨*length*⟩.

`nohead`
: eliminates spaces for the head of the page, which is equivalent to both `\headheight=0pt` and `\headsep=0pt`.

`nofoot`
: eliminates spaces for the foot of the page, which is equivalent to `\footskip=0pt`.

`noheadfoot`
: equivalent to `nohead` and `nofoot`, which means that `\headheight`, `\headsep` and `\footskip` are all set to `0pt`.

`footnotesep`
: changes the dimension `\skip\footins`, separation between the bottom of text body and the top of footnote text.

`marginparwidth` | `marginpar`
: modifies `\marginparwidth`, width of the marginal notes. `marginparwidth=`⟨*length*⟩.

`marginparsep`
: modifies `\marginparsep`, separation between body and marginal notes. `marginparsep=`⟨*length*⟩.

`nomarginpar`
: shrinks spaces for marginal notes to 0pt, which is equivalent to `\marginparwidth=0pt` and `\marginparsep=0pt`.

`columnsep`
: modifies `\columnsep`, the separation between two columns in `twocolumn` mode.

`hoffset`
: modifies `\hoffset`. `hoffset=`⟨*length*⟩.

`voffset`
: modifies `\voffset`. `voffset=`⟨*length*⟩.

`offset`
: horizontal and vertical offset. `offset={`⟨*hoffset*⟩`,`⟨*voffset*⟩`}` or `offset=`⟨*length*⟩.

`twocolumn`
: sets `twocolumn` mode with `\@twocolumntrue`. `twocolumn=false` denotes onecolumn mode with`\@twocolumnfalse`. Instead of `twocolumn=false`, you can specify `onecolumn` (which defaults to `true`)

`onecolumn`
: works as `twocolumn=false`. On the other hand, `onecolumn=false` is equivalent to `twocolumn`.

`twoside`
: sets both `\@twosidetrue` and `\@mparswitchtrue`. See Section 5.4.

`textwidth`
: sets `\textwidth` directly. See Section 5.3.

`textheight`
: sets `\textheight` directly. See Section 5.3.

`reversemp` | `reversemarginpar`
: makes the marginal notes appear in the left (inner) margin with `\@reversemargintrue`. The option doesn't change `includemp` mode. It's set `false` by default.

5.6 Drivers

The package supports drivers `dvips`, `dvipdfm`, `pdftex`, `luatex`, `xetex` and `vtex`. You can also set `dvipdfm` for dvipdfmx and xdvipdfmx The options `dvipdfmx` and `xdvipdfmx` are also supported as aliases for the `dvipdfm` option. `pdftex` for pdflatex, and `vtex` for VTEX environment. The driver options are exclusive. The driver can be set by either `driver=`⟨*driver name*⟩ or any of the drivers directly like `pdftex`. By default, geometry guesses the driver appropriate to the system in use. Therefore, you don't have to set a driver in most cases. However, if you want to use `dvipdfm`, you should specify it explicitly.

† `driver` specifies the driver with `driver=`⟨*driver name*⟩. `dvips`, `dvipdfm`, `pdftex`, `luatex`, `vtex`, `xetex`, `auto` and `none` are available as a driver name. The names except for `auto` and `none` can be specified directly with the name without `driver=`. `driver=auto` makes the auto-detection work whatever the previous setting is. `driver=none` disables the auto-detection and sets no driver, which may be useful when you want to let other package work out the driver setting. For example, if you want to use crop package with geometry, you should call `\usepackage[driver=none]{geometry}` before the crop package.

† `dvips` writes the paper size in dvi output with the `\special` macro. If you use *dvips* as a DVI-to-PS driver, for example, to print a document with `\geometry{a3paper,landscape}` on A3 paper in landscape orientation, you don't need options "`-t a3 -t landscape`" to *dvips*.

† `dvipdfm` works like `dvips` except for landscape correction. You can set this option when using dvipdfmx and xdvipdfmx to process the dvi output.

† `pdftex` sets `\pdfpagewidth` and `\pdfpageheight` internally.

† `luatex` sets `\pagewidth` and `\pageheight` internally.

† `xetex` is the same as `pdftex` except for ignoring `\pdf{h,v}origin` undefined in XeLaTeX. This option is introduced in the version 5. Note that 'geometry.cfg' in TEX Live, which disables the auto-detection routine and sets `pdftex`, is no longer necessary, but has no problem even though it's left undeleted. Instead of `xetex`, you can specify `dvipdfm` with XeLaTeX if you want to use specials of dvipdfm XeTeX supports.

† `vtex` sets dimensions `\mediawidth` and `\mediaheight` for VTEX. When this driver is selected (explicitly or automatically), geometry will auto-detect which output mode (DVI, PDF or PS) is selected in VTEX, and do proper settings for it.

If explicit driver setting is mismatched with the typesetting program in use, the default driver `dvips` would be selected.

5.7 Other options

The other useful options are described here.

† `verbose` displays the parameter results on the terminal. `verbose=false` (default) still puts them into the log file.

† `reset` sets back the layout dimensions and switches to the settings before geometry is loaded. Options given in `geometry.cfg` are also cleared. Note that this cannot reset `pass` and `mag` with `truedimen`. `reset=false` has no effect and cannot cancel the previous `reset`(`=true`) if any. For example, when you go

```
\documentclass[landscape]{article}
\usepackage[twoside,reset,left=2cm]{geometry}
```

with `\ExecuteOptions{scale=0.9}` in `geometry.cfg`, then as a result, `landscape` and `left=2cm` remain effective, and `scale=0.9` and `twoside` are ineffective.

† `mag` sets magnification value (`\mag`) and automatically modifies `\hoffset` and `\voffset` according to the magnification. `mag=`⟨*value*⟩. Note that ⟨*value*⟩ should be an integer value with 1000 as a normal size. For example, `mag=1414` with `a4paper` provides an enlarged print fitting in `a3paper`, which is 1.414 ($=\sqrt{2}$) times larger than `a4paper`. Font enlargement needs extra disk space. **Note that setting `mag` should precede any other settings with 'true' dimensions, such as `1.5truein`, `2truecm` and so on.** See also `truedimen` option.

† `truedimen` changes all internal explicit dimension values into *true* dimensions, e.g., `1in` is changed to `1truein`. Typically this option will be used together with `mag` option. Note that this is ineffective against externally specified dimensions. For example, when you set "`mag=1440, margin=10pt, truedimen`", margins are not 'true' but magnified. If you want to set exact margins, you should set like "`mag=1440, margin=10truept, truedimen`" instead.

† `pass` disables all of the geometry options and calculations except `verbose` and `showframe`. It is order-independent and can be used for checking out the page layout of the documentclass, other packages and manual settings without geometry.

† `showframe` shows visible frames for the text area and page, and the lines for the head and foot on the first page.

† `showcrop` prints crop marks at each corner of user-specified layout area.

6 Processing options

6.1 Order of loading

If there's `geometry.cfg` somewhere TEX can find it, geometry loads it first. For example, in `geometry.cfg` you may write `\ExecuteOptions{a4paper}`, which specifies A4 paper as the default paper. Basically you can use all the options defined in geometry with `\ExecuteOptions{}`.

The order of loading in the preamble of your document is as follows:

1. `geometry.cfg` if it exists.
2. Options specified with `\documentclass[`⟨*options*⟩`]{...}`.
3. Options specified with `\usepackage[`⟨*options*⟩`]{geometry}`
4. Options specified with `\geometry{`⟨*options*⟩`}`, which can be called multiple times. (`reset` option will cancel the specified options ever given in `\usepackage{geometry}` or `\geometry`.)

6.2 Order of options

The specification of geometry options is order-independent, and overwrites the previous one for the same setting. For example,

`[left=2cm, right=3cm]` is equivalent to `[right=3cm, left=2cm]`.

The options called multiple times overwrite the previous settings. For example,

`[verbose=true, verbose=false]` results in `verbose=false`.

`[hmargin={3cm,2cm}, left=1cm]` is the same as `hmargin={1cm,2cm}`, where the left (or inner) margin is overwritten by `left=1cm`.

`reset` and `mag` are exceptions. The `reset` option removes all the geometry options (except `pass`) before it. If you set

```
\documentclass[landscape]{article}
\usepackage[margin=1cm,twoside]{geometry}
\geometry{a5paper, reset, left=2cm}
```

then `margin=1cm`, `twoside` and `a5paper` are removed, and is eventually equivalent to

```
\documentclass[landscape]{article}
\usepackage[left=2cm]{geometry}
```

The `mag` option should be set in advance of any other settings with 'true' length, such as `left=1.5truecm`, `width=5truein` and so on. The `\mag` primitive can be set before this package is called.

6.3 Priority

There are several ways to set dimensions of the *body*: `scale`, `total`, `text` and `lines`. The geometry package gives higher priority to the more concrete specification. Here is the priority rule for *body*.

$$\text{priority:} \quad \text{low} \longrightarrow \text{high}$$

$$\left\{\begin{array}{l}\texttt{hscale}\\ \texttt{vscale}\\ \texttt{scale}\end{array}\right\} < \left\{\begin{array}{l}\texttt{width}\\ \texttt{height}\\ \texttt{total}\end{array}\right\} < \left\{\begin{array}{l}\texttt{textwidth}\\ \texttt{textheight}\\ \texttt{text}\end{array}\right\} < \texttt{lines}.$$

For example,

```
\usepackage[hscale=0.8, textwidth=7in, width=18cm]{geometry}
```

is the same as `\usepackage[textwidth=7in]{geometry}`. Another example:

```
\usepackage[lines=30, scale=0.8, text=7in]{geometry}
```

results in `[lines=30, textwidth=7in]`.

6.4 Defaults

This section sums up the default settings for the auto-completion described later.

The default vertical margin ratio is 2/3, namely,

$$\texttt{top} : \texttt{bottom} = 2 : 3 \qquad \textit{default}. \tag{6}$$

As for the horizontal margin ratio, the default value depends on whether the document is onesided or twosided,

$$\texttt{left}\ (\texttt{inner}) : \texttt{right}\ (\texttt{outer}) = \left\{\begin{array}{ll} 1:1 & \textit{default for oneside},\\ 2:3 & \textit{default for twoside}.\end{array}\right. \tag{7}$$

Obviously the default horizontal margin ratio for oneside is 'centering'.

The geometry package has the following default setting for *onesided* documents:

- `scale=0.7` (*body* is $0.7 \times$ *paper*)
- `marginratio={1:1, 2:3}` (1:1 for horizontal and 2:3 for vertical margins)
- `ignoreall` (the header, footer, marginal notes are excluded when calculating the size of *body*.)

For *twosided* document with `twoside` option, the default setting is the same as *onesided* except that the horizontal margin ratio is set to `2:3` as well.

Additional options overwrite the previous specified dimensions.

6.5 Auto-completion

Figure 7 shows schematically how many specification patterns exist and how to solve the ambiguity of the specifications. Each axis shows the numbers of lengths explicitly specified for body and margins. S(m,b) presents the specification with a set of numbers (*margin*, *body*) = (m, b).

For example, the specification `width=14cm, left=3cm` is categorized into S(1,1), which is an adequate specification. If you add `right=4cm`, it would be in S(2,1) and overspecified. If only `width=14cm` is given, it's in S(0,1), underspecified.

The geometry package has the auto-completion mechanism, in which if the layout parameters are underspecified or overspecified, geometry works out the ambiguity using the defaults and other relations. Here are the specifications and the completion rules.

S(0,0) Nothing is specified. The geometry package sets *body* with the default `scale` (= 0.7). For example, `width` is set to be `0.7` $\times$ `layoutwidth`. Note that by default `layoutwidth` and `layoutheight` will be equal to `\paperwidth` and `\paperheight` respectively. Thus S(0,0) goes to S(0,1). See S(0,1).

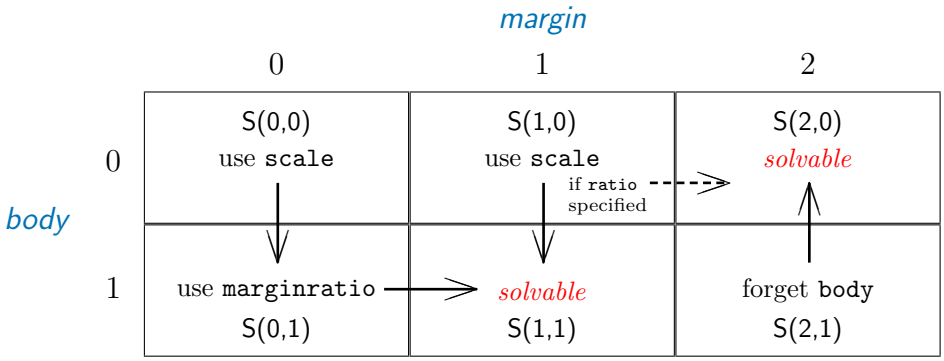


Figure 7: Specifications S(0,0) to S(2,1) and the completion rules (arrows). Column and row numbers denote the number of explicitly specified lengths for margin and body respectively. S(m,b) denote a specification with a set of the numbers (*margin*, *body*) = (m, b).

S(0,1) Only *body* is specified, such as `width=7in`, `lines=20`, `body={20cm,24cm}`, `scale=0.9` and so forth. Then geometry sets margins with the margin ratio. If the margin ratio is not specified, the default is used. The default vertical margin ratio is defined as

$$\texttt{top} : \texttt{bottom} = 2 : 3 \qquad \textit{default}. \tag{8}$$

As for the horizontal margin ratio, the default value depends on whether the document is onesided or twosided,

$$\texttt{left}\ (\texttt{inner}) : \texttt{right}\ (\texttt{outer}) = \begin{cases} 1:1 & \textit{default for oneside}, \\ 2:3 & \textit{default for twoside}. \end{cases} \tag{9}$$

For example, if `height=22cm` is specified on A4 paper, geometry calculates `top` margin as follows:

$$\begin{aligned} \texttt{top} &= (\texttt{layoutheight} - \texttt{height}) \times 2/5 \\ &= (29.7 - 22) \times 2/5 = 3.08(\text{cm}) \end{aligned} \tag{10}$$

Thus `top` margin and body `height` have been determined, the specification for the vertical goes to S(1,1) and all the parameters can be solved.

S(1,0) Only one margin is specified, such as `bottom=2cm`, `left=1in`, `top=3cm`, and so forth.

- **If the margin ratio is *not* specified**, geometry sets *body* with the default `scale` (= 0.7). For example, if `top=2.4cm` is specified, geometry sets

  $$\texttt{height} = 0.7 \times \texttt{layoutheight} \quad (= 0.7\texttt{\textbackslash paperheight by default}),$$

  then S(1,0) goes to S(1,1), in which `bottom` is calculated with `layoutheight` − (`height` + `top`) and results in 6.51cm on A4 paper if the layout size is equal to the paper size.

- **If the margin ratio is specified**, such as `hmarginratio={1:2}`, `vratio={3:4}` and so forth, geometry sets the other margin with the specified margin ratio. For example, if a set of options "`top=2.4cm,vratio={3:4}`" is specified, geometry sets `bottom` to be `3.2cm` calculating

  $$\texttt{bottom} = \texttt{top}/3 \times 4 = 3.2\text{cm}$$

  Thus S(1,0) goes to S(2,0).

Note that the version 4 or earlier used to set the other margin with the margin ratio. In the version 5, therefore, with the same specification, the result will be different from the one in the version 4. For example, if only `top=2.4cm` is specified, you got `bottom=2.4cm` in the version 4 or earlier, but you will get `bottom=6.51cm` in the version 5.

S(2,1) The *body* and two *margins* are all specified, such as `vdivide={1in,8in,1.5in}`, "`left=3cm,width=13cm,right=4cm`" and so forth. Since geometry basically gives priority to *margins* if dimensions are overspecified, geometry forgets and resets *body*. For example, if you specify

```
\usepackage[a4paper,left=3cm,width=13cm,right=4cm]{geometry},
```

`width` is reset to be 14cm because the width of a A4 paper is 21cm long.

7 Changing layout mid-document

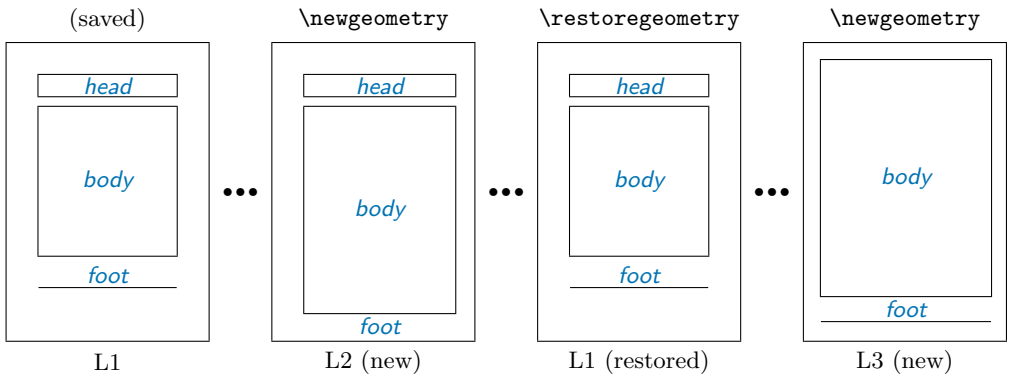
The version 5 provides the new commands `\newgeometry{⋯}` and `\restoregeometry`, which allow you to change page dimensions in the middle of the document. Unlike `\geometry` in the preamble, `\newgeometry` is available only after `\begin{document}`, resets all the options ever specified except for the papersize-related options: `landscape`, `portrait`, and paper size options (such as `papersize`, `paper=a4paper` and so forth), which can't be changed with `\newgeometry`.

The command `\restoregeometry` restores the page layout specified in the preamble (before `\begin{document}`) with the options to `\usepackage{geometry}` and `\geometry`.

Note that both `\newgeometry` and `\restoregeometry` insert `\clearpage` where they are called.

Below is an example of changing layout mid-document. The layout L1 specified with `hmargin=3cm` (`left` and `right` margins are `3cm` long) is changed to L2 with `left=3cm`, `right=1cm` and `bottom=0.1cm`. The layout L1 is restored with `\restoregeometry`.

```
\usepackage[hmargin=3cm]{geometry}
\begin{document}
      Layout L1
\newgeometry{left=3cm,right=1cm,bottom=0.1cm}
      Layout L2 (new)
\restoregeometry
      Layout L1 (restored)
\newgeometry{margin=1cm,includefoot}
      Layout L3 (new)
\end{document}
```



A set of commands `\savegeometry{⟨name⟩}` and `\loadgeometry{⟨name⟩}` is handy if you want to reuse more different layouts in your document. For example,

```
\usepackage[hmargin=3cm]{geometry}
\begin{document}
      L1
\newgeometry{left=3cm,right=1cm,bottom=0.1cm}
\savegeometry{L2}
      L2 (new, saved)
```

```
\restoregeometry
      L1 (restored)
\newgeometry{margin=1cm,includefoot}
      L3 (new)
\loadgeometry{L2}
      L2 (loaded)
\end{document}
```

8 Examples

1. A onesided page layout with the text area centered in the paper. The examples below have the same result because the horizontal margin ratio is set `1:1` for oneside by default.

   - `centering`
   - `marginratio=1:1`
   - `vcentering`

2. A twosided page layout with the inside offset for binding set to `1cm`.

   - `twoside, bindingoffset=1cm`

   In this case, `textwidth` is shorter than that of the default twosided document by $0.7\times$`1cm` (= `0.7cm`) because the default width of *body* is set with `scale=0.7`, which means `width` = `0.7` × `layoutwidth` (= `0.7\paperwidth` by default).

3. A layout with the left, right, and top margin `3cm`, `2cm` and `2.5in` respectively, with textheight of 40 lines, and with the head and foot of the page included in *total body*. The two examples below have the same result.

   - `left=3cm, right=2cm, lines=40, top=2.5in, includeheadfoot`
   - `hmargin={3cm,2cm}, tmargin=2.5in, lines=40, includeheadfoot`

4. A layout with the height of *total body* `10in`, the bottom margin `2cm`, and the default width. The top margin will be calculated automatically. Each solution below results in the same page layout.

   - `vdivide={*, 10in, 2cm}`
   - `bmargin=2cm, height=10in`
   - `bottom=2cm, textheight=10in`

   Note that dimensions for *head* and *foot* are excluded from `height` of *total body*. An additional `includefoot` makes `\footskip` included in `totalheight`. Therefore, in the two cases below, `textheight` in the former layout is shorter than the latter (with 10in exactly) by `\footskip`. In other words, `height` = `textheight` + `footskip` when `includefoot=true` in this case.

   - `bmargin=2cm, height=10in, includefoot`
   - `bottom=2cm, textheight=10in, includefoot`

5. A layout with textwidth and textheight 90% of the paper and with *body* centered. Each solution below results in the same page layout as long as `layoutwidth` and `layoutheight` are not modified from the default.

   - `scale=0.9, centering`
   - `text={.9\paperwidth,.9\paperheight}, ratio=1:1`
   - `width=.9\paperwidth, vmargin=.05\paperheight, marginratio=1:1`
   - `hdivide={*,0.9\paperwidth,*}, vdivide={*,0.9\paperheight,*}` (as for onesided documents)
   - `margin={0.05\paperwidth,0.05\paperheight}`

   You can add `heightrounded` to avoid an "underfull vbox warning" like

   ```
   Underfull \vbox (badness 10000) has occurred while \output is active.
   ```

See Section 5.3 for the detailed description about `heightrounded`.

6. A layout with the width of marginal notes set to `3cm` and included in the width of *total body*. The following examples are the same.
   - `marginparwidth=3cm, includemp`
   - `marginpar=3cm, ignoremp=false`
7. A layout where *body* occupies the whole paper with A5 paper in landscape. The following examples are the same.
   - `a5paper, landscape, scale=1.0`
   - `landscape=TRUE, paper=a5paper, margin=0pt`
8. A screen size layout appropriate for presentation with PC and video projector.

```
\documentclass{slide}
\usepackage[screen,margin=0.8in]{geometry}
 ...
\begin{slide}
   ...
\end{slide}
```

9. A layout with fonts and spaces both enlarged from A4 to A3. In the case below, the resulting paper size is A3.
   - `a4paper, mag=1414`.

   If you want to have a layout with two times bigger fonts, but without changing paper size, you can type
   - `letterpaper, mag=2000, truedimen`.

   You can add `dvips` option, that is useful to preview it with proper paper size by `dviout` or `xdvi`.
10. Changing the layout of the first page and leaving the others as default before loading geometry. Use `pass` option, `\newgeometry` and `\restoregeometry`.

```
\documentclass{book}
\usepackage[pass]{geometry}
   % 'pass' disregards the package layout,
   %  so the original 'book' layout is memorized here.
\begin{document}
\newgeometry{margin=1cm}% changes the first page dimensions.
   Page 1
\restoregeometry % restores the original 'book' layout.
   Page 2 and more
\end{document}
```

11. A complex page layout.

```
\usepackage[a5paper, landscape, twocolumn, twoside,
    left=2cm, hmarginratio=2:1, includemp, marginparwidth=43pt,
    bottom=1cm, foot=.7cm, includefoot, textheight=11cm, heightrounded,
    columnsep=1cm, dvips,  verbose]{geometry}
```

    Try typesetting it and checking out the result yourself. `:-)`

9 Known problems

- With `mag` $\neq 1000$ and `truedimen`, `paperwidth` and `paperheight` shown in verbose mode are different from the real size of the resulted PDF. The PDF itself is correct anyway.
- With `mag` $\neq 1000$, *no* `truedimen` and hyperref, hyperref should be loaded before geometry. Otherwise the resulted PDF size will become wrong.
- With crop package and `mag` $\neq 1000$, `center` option of crop doesn't work well.

10 Acknowledgments

The author appreciates helpful suggestions and comments from many people including (in alphabetical order of family name): Jean-Bernard Addor, Frank Bennett, Alexis Dimitriadis, Friedrich Flender, Adrian Heathcote, Stephan Hennig, Morten Høgholm, Jonathan Kew, James Kilfiger, Yusuke Kuroki, Jean-Marc Lasgouttes, Wlodzimierz Macewicz, Frank Mittelbach, Eckhard Neber, Rolf Niepraschk, Hans Fr. Nordhaug, Heiko Oberdiek, Keith Reckdahl, Peter Riocreux, Will Robertson, Pablo Rodriguez, Nico Schlöemer, Perry C. Stearns, Frank Stengel, Plamen Tanovski, Petr Uher, Piet van Oostrum, Vladimir Volovich, Michael Vulis.

11 Implementation

```
1 ⟨*package⟩
```

This package requires the following packages: keyval, ifvtex.

```
2 \RequirePackage{keyval}%
3 \RequirePackage{ifvtex}%
```

Internal switches are declared here.

```
4 \newif\ifGm@verbose
5 \newif\ifGm@landscape
6 \newif\ifGm@swap@papersize
7 \newif\ifGm@includehead
8 \newif\ifGm@includefoot
9 \newif\ifGm@includemp
10 \newif\ifGm@hbody
11 \newif\ifGm@vbody
12 \newif\ifGm@heightrounded
13 \newif\ifGm@showframe
14 \newif\ifGm@showcrop
15 \newif\ifGm@pass
16 \newif\ifGm@resetpaper
17 \newif\ifGm@layout
18 \newif\ifGm@newgm
```

`\Gm@cnth` `\Gm@cntv` The counters for horizontal and vertical partitioning patterns.

```
19 \newcount\Gm@cnth
20 \newcount\Gm@cntv
```

`\c@Gm@tempcnt` The counter is used to set number with calc.

```
21 \newcount\c@Gm@tempcnt
```

`\Gm@bindingoffset` The binding offset for the inner margin.

```
22 \newdimen\Gm@bindingoffset
```

`\Gm@wd@mp` `\Gm@odd@mp` `\Gm@even@mp` Correction lengths for `\textwidth`, `\oddsidemargin` and `\evensidemargin` in `includemp` mode.

```
23 \newdimen\Gm@wd@mp
24 \newdimen\Gm@odd@mp
25 \newdimen\Gm@even@mp
```

`\Gm@layoutwidth` `\Gm@layoutheight` `\Gm@layouthoffset` `\Gm@layoutvoffset` The dimensions for the layout area.

```
26 \newdimen\Gm@layoutwidth
27 \newdimen\Gm@layoutheight
28 \newdimen\Gm@layouthoffset
29 \newdimen\Gm@layoutvoffset
```

`\Gm@dimlist` The token in which LaTeX native dimensions can be stored.

```
30 \newtoks\Gm@dimlist
```

`\Gm@warning` The macro to print warning messages.

```
31 \def\Gm@warning#1{\PackageWarningNoLine{geometry}{#1}}%
```

`\ifGm@preamble` The macro executes the option given as an argument only if it's specified in the preamble, as the options of `\usepackage` and/or the argument of `\geometry`. Otherwise, the macro would print the warning message and ignores the option setting.

```
32 \def\ifGm@preamble#1{%
33   \ifGm@newgm
34    \Gm@warning{‘#1’: not available in ‘\string\newgeometry’; skipped}%
35   \else
36     \expandafter\@firstofone
37   \fi}%
```

`\Gm@Dhratio` `\Gm@Dhratiotwo` `\Gm@Dvratio` The default values for the horizontal and vertical *marginalratio* are defined. `\Gm@Dhratiotwo` denotes the default value of horizontal *marginratio* for twoside page layout with left and right margins swapped on verso pages, which is set by `twoside`.

```
38 \def\Gm@Dhratio{1:1}% = left:right default for oneside
39 \def\Gm@Dhratiotwo{2:3}% = inner:outer default for twoside.
40 \def\Gm@Dvratio{2:3}% = top:bottom default
```

\Gm@Dhscale
\Gm@Dvscale

The default values for the horizontal and vertical *scale* are defined with 0.7.

```
\def\Gm@Dhscale{0.7}%
\def\Gm@Dvscale{0.7}%
```

\Gm@dvips
\Gm@dvipdfm
\Gm@pdftex
\Gm@luatex
\Gm@xetex
\Gm@vtex

The driver names.

```
\def\Gm@dvips{dvips}%
\def\Gm@dvipdfm{dvipdfm}%
\def\Gm@pdftex{pdftex}%
\def\Gm@luatex{luatex}%
\def\Gm@xetex{xetex}%
\def\Gm@vtex{vtex}%
```

\Gm@true
\Gm@false

The macros for `true` and `false`.

```
\def\Gm@true{true}%
\def\Gm@false{false}%
```

\Gm@orgpw
\Gm@orgph

These macros keep original paper (media) size intact.

```
\edef\Gm@orgpw{\the\paperwidth}%
\edef\Gm@orgph{\the\paperheight}%
```

\Gm@savelength

The macro saves the specified length to `\Gm@restore`.

```
\def\Gm@savelength#1{%
  \g@addto@macro\Gm@restore{\expandafter\noexpand\expandafter\csname
  #1\endcsname\expandafter=\expandafter\the\csname #1\endcsname\relax}}%
```

\Gm@saveboolean

The macro saves the specified boolean to `\Gm@restore`.

```
\def\Gm@saveboolean#1{%
  \csname if#1\endcsname
    \g@addto@macro\Gm@restore{\expandafter\noexpand\csname #1true\endcsname}%
  \else
    \g@addto@macro\Gm@restore{\expandafter\noexpand\csname #1false\endcsname}%
  \fi}%
```

\Gm@restore

The initialization for `\Gm@restore`.

```
\def\Gm@restore{}%
```

\Gm@save

The definition of the macro saving the real lengths LaTeX options.

```
\def\Gm@save{%
  \Gm@savelength{paperwidth}%
  \Gm@savelength{paperheight}%
  \Gm@savelength{textwidth}%
  \Gm@savelength{textheight}%
  \Gm@savelength{evensidemargin}%
  \Gm@savelength{oddsidemargin}%
  \Gm@savelength{topmargin}%
  \Gm@savelength{headheight}%
  \Gm@savelength{headsep}%
  \Gm@savelength{topskip}%
  \Gm@savelength{footskip}%
  \Gm@savelength{baselineskip}%
  \Gm@savelength{marginparwidth}%
  \Gm@savelength{marginparsep}%
  \Gm@savelength{columnsep}%
  \Gm@savelength{hoffset}%
  \Gm@savelength{voffset}
  \Gm@savelength{Gm@layoutwidth}%
  \Gm@savelength{Gm@layoutheight}%
  \Gm@savelength{Gm@layouthoffset}%
  \Gm@savelength{Gm@layoutvoffset}%
  \Gm@saveboolean{@twocolumn}%
  \Gm@saveboolean{@twoside}%
  \Gm@saveboolean{@mparswitch}%
  \Gm@saveboolean{@reversemargin}}%
```

`\Gm@initnewgm` The macro initializes the parameters for layout in `\newgeometry`.

```
\def\Gm@initnewgm{%
  \Gm@passfalse
  \Gm@swap@papersizefalse
  \Gm@dimlist={}
  \Gm@hbodyfalse
  \Gm@vbodyfalse
  \Gm@heightroundedfalse
  \Gm@includeheadfalse
  \Gm@includefootfalse
  \Gm@includempfalse
  \let\Gm@width\@undefined
  \let\Gm@height\@undefined
  \let\Gm@textwidth\@undefined
  \let\Gm@textheight\@undefined
  \let\Gm@lines\@undefined
  \let\Gm@hscale\@undefined
  \let\Gm@vscale\@undefined
  \let\Gm@hmarginratio\@undefined
  \let\Gm@vmarginratio\@undefined
  \let\Gm@lmargin\@undefined
  \let\Gm@rmargin\@undefined
  \let\Gm@tmargin\@undefined
  \let\Gm@bmargin\@undefined
  \Gm@layoutfalse
  \Gm@layouthoffset\z@
  \Gm@layoutvoffset\z@
  \Gm@bindingoffset\z@}%
```

`\Gm@initall` This initialization is called as soon as the package is load It's also called as soon as `reset` option is specified.

```
\def\Gm@initall{%
  \let\Gm@driver\@empty
  \let\Gm@truedimen\@empty
  \let\Gm@paper\@undefined
  \Gm@resetpaperfalse
  \Gm@landscapefalse
  \Gm@verbosefalse
  \Gm@showframefalse
  \Gm@showcropfalse
  \Gm@newgmfalse
  \Gm@initnewgm}%
```

`\Gm@setdriver` The macro sets the specified driver.

```
\def\Gm@setdriver#1{%
  \expandafter\let\expandafter\Gm@driver\csname Gm@#1\endcsname}%
```

`\Gm@unsetdriver` The macro unsets the specified driver if it has been set.

```
\def\Gm@unsetdriver#1{%
  \expandafter\ifx\csname Gm@#1\endcsname\Gm@driver\let\Gm@driver\@empty\fi}%
```

`\Gm@setbool` `\Gm@setboolrev` The macros for boolean option processing.

```
\def\Gm@setbool{\@dblarg\Gm@@setbool}%
\def\Gm@setboolrev{\@dblarg\Gm@@setboolrev}%
\def\Gm@@setbool[#1]#2#3{\Gm@doif{#1}{#3}{\csname Gm@#2\Gm@bool\endcsname}}%
\def\Gm@@setboolrev[#1]#2#3{\Gm@doifelse{#1}{#3}%
  {\csname Gm@#2\Gm@false\endcsname}{\csname Gm@#2\Gm@true\endcsname}}%
```

`\Gm@doif` `\Gm@doifelse` `\Gm@doif` excutes the third argument `#3` using a boolean value `#2` of a option `#1`. `\Gm@doifelse` executes the third argument `#3` if a boolean option `#1` with its value `#2` `true`, and executes the fourth argument `#4` if `false`.

```
\def\Gm@doif#1#2#3{%
  \lowercase{\def\Gm@bool{#2}}%
  \ifx\Gm@bool\@empty
```

```
    \let\Gm@bool\Gm@true
  \fi
  \ifx\Gm@bool\Gm@true
  \else
    \ifx\Gm@bool\Gm@false
    \else
      \let\Gm@bool\relax
    \fi
  \fi
  \ifx\Gm@bool\relax
    \Gm@warning{‘#1’ should be set to ‘true’ or ‘false’}%
  \else
    #3
  \fi}%
\def\Gm@doifelse#1#2#3#4{%
  \Gm@doif{#1}{#2}{\ifx\Gm@bool\Gm@true #3\else #4\fi}}%
```

`\Gm@reverse` The macro reverses a bool value.

```
\def\Gm@reverse#1{%
  \csname ifGm@#1\endcsname
  \csname Gm@#1false\endcsname\else\csname Gm@#1true\endcsname\fi}%
```

`\Gm@defbylen` `\Gm@defbycnt` Macros `\Gm@defbylen` and `\Gm@defbycnt` can be used to define `\Gm@xxxx` variables by length and counter respectively with calc package.

```
\def\Gm@defbylen#1#2{%
  \begingroup\setlength\@tempdima{#2}%
  \expandafter\xdef\csname Gm@#1\endcsname{\the\@tempdima}\endgroup}%
\def\Gm@defbycnt#1#2{%
  \begingroup\setcounter{Gm@tempcnt}{#2}%
  \expandafter\xdef\csname Gm@#1\endcsname{\the\value{Gm@tempcnt}}\endgroup}%
```

`\Gm@set@ratio` The macro parses the value of options specifying marginal ratios, which is used in `\Gm@setbyratio` macro.

```
\def\Gm@sep@ratio#1:#2{\@tempcnta=#1\@tempcntb=#2}%
```

`\Gm@setbyratio` The macro determines the dimension specified by `#4` calculating `#3`$\times a/b$, where a and b are given by `\Gm@mratio` with $a : b$ value. If `#1` in brackets is `b`, a and b are swapped. The second argument with `h` or `v` denoting horizontal or vertical is not used in this macro.

```
\def\Gm@setbyratio[#1]#2#3#4{% determine #4 by ratio
  \expandafter\Gm@sep@ratio\Gm@mratio\relax
  \if#1b
    \edef\@@tempa{\the\@tempcnta}%
    \@tempcnta=\@tempcntb
    \@tempcntb=\@@tempa\relax
  \fi
  \expandafter\setlength\expandafter\@tempdimb\expandafter
    {\csname Gm@#3\endcsname}%
  \ifnum\@tempcntb>\z@
    \multiply\@tempdimb\@tempcnta
    \divide\@tempdimb\@tempcntb
  \fi
  \expandafter\edef\csname Gm@#4\endcsname{\the\@tempdimb}}%
```

`\Gm@detiv` This macro determines the fourth length(`#4`) from `#1`(layoutwidth or layoutheight), `#2` and `#3`. It is used in `\Gm@detall` macro.

```
\def\Gm@detiv#1#2#3#4{% determine #4.
  \expandafter\setlength\expandafter\@tempdima\expandafter
    {\csname Gm@layout#1\endcsname}%
  \expandafter\setlength\expandafter\@tempdimb\expandafter
    {\csname Gm@#2\endcsname}%
  \addtolength\@tempdima{-\@tempdimb}%
  \expandafter\setlength\expandafter\@tempdimb\expandafter
    {\csname Gm@#3\endcsname}%
  \addtolength\@tempdima{-\@tempdimb}%
```

```
  \ifdim\@tempdima<\z@
    \Gm@warning{‘#4’ results in NEGATIVE (\the\@tempdima).%
    ^^J\@spaces ‘#2’ or ‘#3’ should be shortened in length}%
  \fi
  \expandafter\edef\csname Gm@#4\endcsname{\the\@tempdima}}%
```

`\Gm@detiiandiii` This macro determines `#2` and `#3` from `#1` with the first argument (`#1`) can be `width` or `height`, which is expanded into dimensions of paper and total body. It is used in `\Gm@detall` macro.

```
\def\Gm@detiiandiii#1#2#3{% determine #2 and #3.
  \expandafter\setlength\expandafter\@tempdima\expandafter
    {\csname Gm@layout#1\endcsname}%
  \expandafter\setlength\expandafter\@tempdimb\expandafter
    {\csname Gm@#1\endcsname}%
  \addtolength\@tempdima{-\@tempdimb}%
  \ifdim\@tempdima<\z@
    \Gm@warning{‘#2’ and ‘#3’ result in NEGATIVE (\the\@tempdima).%
                  ^^J\@spaces ‘#1’ should be shortened in length}%
  \fi
  \ifx\Gm@mratio\@undefined
    \expandafter\Gm@sep@ratio\Gm@Dmratio\relax
  \else
    \expandafter\Gm@sep@ratio\Gm@mratio\relax
    \ifnum\@tempcntb>\z@\else
      \Gm@warning{margin ratio a:b should be non-zero; default used}%
      \expandafter\Gm@sep@ratio\Gm@Dmratio\relax
    \fi
  \fi
  \@tempdimb=\@tempdima
  \advance\@tempcntb\@tempcnta
  \divide\@tempdima\@tempcntb
  \multiply\@tempdima\@tempcnta
  \advance\@tempdimb-\@tempdima
  \expandafter\edef\csname Gm@#2\endcsname{\the\@tempdima}%
  \expandafter\edef\csname Gm@#3\endcsname{\the\@tempdimb}}%
```

`\Gm@detall` This macro determines partition of each direction. The first argument (`#1`) should be `h` or `v`, the second (`#2`) `width` or `height`, the third (`#3`) `lmargin` or `top`, and the last (`#4`) `rmargin` or `bottom`.

```
\def\Gm@detall#1#2#3#4{%
  \@tempcnta\z@
  \if#1h
    \let\Gm@mratio\Gm@hmarginratio
    \edef\Gm@Dmratio{\if@twoside\Gm@Dhratiotwo\else\Gm@Dhratio\fi}%
  \else
    \let\Gm@mratio\Gm@vmarginratio
    \edef\Gm@Dmratio{\Gm@Dvratio}%
  \fi
```

`\@tempcnta` is treated as a three-digit binary value with top, middle and bottom denoted `left`(`top`), `width`(`height`) and `right`(`bottom`) margins user specified respectively.

```
  \if#1h
    \ifx\Gm@lmargin\@undefined\else\advance\@tempcnta4\relax\fi
    \ifGm@hbody\advance\@tempcnta2\relax\fi
    \ifx\Gm@rmargin\@undefined\else\advance\@tempcnta1\relax\fi
    \Gm@cnth\@tempcnta
  \else
    \ifx\Gm@tmargin\@undefined\else\advance\@tempcnta4\relax\fi
    \ifGm@vbody\advance\@tempcnta2\relax\fi
    \ifx\Gm@bmargin\@undefined\else\advance\@tempcnta1\relax\fi
    \Gm@cntv\@tempcnta
  \fi
```

Case the value is `000` (=0) with nothing fixed (default):

```
  \ifcase\@tempcnta
    \if#1h
      \Gm@defbylen{width}{\Gm@Dhscale\Gm@layoutwidth}%
    \else
```

```
      \Gm@defbylen{height}{\Gm@Dvscale\Gm@layoutheight}%
    \fi
    \Gm@detiiandiii{#2}{#3}{#4}%
```
Case 001 (=1) with `right`(`bottom`) fixed:
```
  \or
    \ifx\Gm@mratio\@undefined
      \if#1h
        \Gm@defbylen{width}{\Gm@Dhscale\Gm@layoutwidth}%
      \else
        \Gm@defbylen{height}{\Gm@Dvscale\Gm@layoutheight}%
      \fi
      \setlength\@tempdimc{\@nameuse{Gm@#4}}%
      \Gm@detiiandiii{#2}{#3}{#4}%
      \expandafter\let\csname Gm@#2\endcsname\@undefined
      \Gm@defbylen{#4}{\@tempdimc}%
    \else
      \Gm@setbyratio[f]{#1}{#4}{#3}%
    \fi
    \Gm@detiv{#2}{#3}{#4}{#2}%
```
Case 010 (=2) with `width`(`height`) fixed:
```
  \or\Gm@detiiandiii{#2}{#3}{#4}%
```
Case 011 (=3) with both `width`(`height`) and `right`(`bottom`) fixed:
```
  \or\Gm@detiv{#2}{#2}{#4}{#3}%
```
Case 100 (=4) with `left`(`top`) fixed:
```
  \or
    \ifx\Gm@mratio\@undefined
      \if#1h
        \Gm@defbylen{width}{\Gm@Dhscale\Gm@layoutwidth}%
      \else
        \Gm@defbylen{height}{\Gm@Dvscale\Gm@layoutheight}%
      \fi
      \setlength\@tempdimc{\@nameuse{Gm@#3}}%
      \Gm@detiiandiii{#2}{#4}{#3}%
      \expandafter\let\csname Gm@#2\endcsname\@undefined
      \Gm@defbylen{#3}{\@tempdimc}%
    \else
      \Gm@setbyratio[b]{#1}{#3}{#4}%
    \fi
    \Gm@detiv{#2}{#3}{#4}{#2}%
```
Case 101 (=5) with both `left`(`top`) and `right`(`bottom`) fixed:
```
  \or\Gm@detiv{#2}{#3}{#4}{#2}%
```
Case 110 (=6) with both `left`(`top`) and `width`(`height`) fixed:
```
  \or\Gm@detiv{#2}{#2}{#3}{#4}%
```
Case 111 (=7) with all fixed though it is over-specified:
```
  \or\Gm@warning{Over-specification in `#1'-direction.%
                  ^^J\@spaces `#2' (\@nameuse{Gm@#2}) is ignored}%
    \Gm@detiv{#2}{#3}{#4}{#2}%
  \else\fi}%
```

`\Gm@clean` The macro for setting unspecified dimensions to be `\@undefined`. This is used by `\geometry` macro.
```
\def\Gm@clean{%
  \ifnum\Gm@cnth<4\let\Gm@lmargin\@undefined\fi
  \ifodd\Gm@cnth\else\let\Gm@rmargin\@undefined\fi
  \ifnum\Gm@cntv<4\let\Gm@tmargin\@undefined\fi
  \ifodd\Gm@cntv\else\let\Gm@bmargin\@undefined\fi
  \ifGm@hbody\else
    \let\Gm@hscale\@undefined
    \let\Gm@width\@undefined
    \let\Gm@textwidth\@undefined
  \fi
  \ifGm@vbody\else
```

```
    \let\Gm@vscale\@undefined
    \let\Gm@height\@undefined
    \let\Gm@textheight\@undefined
  \fi
  }%
```

`\Gm@parse@divide` The macro parses (h,v)`divide` options.

```
\def\Gm@parse@divide#1#2#3#4{%
  \def\Gm@star{*}%
  \@tempcnta\z@
  \@for\Gm@tmp:=#1\do{%
    \expandafter\KV@@sp@def\expandafter\Gm@frag\expandafter{\Gm@tmp}%
    \edef\Gm@value{\Gm@frag}%
    \ifcase\@tempcnta\relax\edef\Gm@key{#2}%
      \or\edef\Gm@key{#3}%
      \else\edef\Gm@key{#4}%
    \fi
    \@nameuse{Gm@set\Gm@key false}%
    \ifx\empty\Gm@value\else
    \ifx\Gm@star\Gm@value\else
      \setkeys{Gm}{\Gm@key=\Gm@value}%
    \fi\fi
    \advance\@tempcnta\@ne}%
  \let\Gm@star\relax}%
```

`\Gm@branch` The macro splits a value into the same two values.

```
\def\Gm@branch#1#2#3{%
  \@tempcnta\z@
  \@for\Gm@tmp:=#1\do{%
    \KV@@sp@def\Gm@frag{\Gm@tmp}%
    \edef\Gm@value{\Gm@frag}%
    \ifcase\@tempcnta\relax% cnta == 0
      \setkeys{Gm}{#2=\Gm@value}%
    \or% cnta == 1
      \setkeys{Gm}{#3=\Gm@value}%
    \else\fi
    \advance\@tempcnta\@ne}%
  \ifnum\@tempcnta=\@ne
    \setkeys{Gm}{#3=\Gm@value}%
  \fi}%
```

`\Gm@magtooffset` This macro is used to adjust offsets by `\mag`.

```
\def\Gm@magtooffset{%
  \@tempdima=\mag\Gm@truedimen sp%
  \@tempdimb=1\Gm@truedimen in%
  \divide\@tempdimb\@tempdima
  \multiply\@tempdimb\@m
  \addtolength{\hoffset}{1\Gm@truedimen in}%
  \addtolength{\voffset}{1\Gm@truedimen in}%
  \addtolength{\hoffset}{-\the\@tempdimb}%
  \addtolength{\voffset}{-\the\@tempdimb}}%
```

`\Gm@setlength` This macro stores LaTeX native dimensions, which are stored and set afterwards.

```
\def\Gm@setlength#1#2{%
  \let\Gm@len=\relax\let\Gm@td=\relax
  \edef\addtolist{\noexpand\Gm@dimlist=%
  {\the\Gm@dimlist \Gm@len{#1}{#2}}}\addtolist}%
```

`\Gm@expandlengths` This macro processes `\Gm@dimlist`.

```
\def\Gm@expandlengths{%
  \def\Gm@td{\Gm@truedimen}%
  \def\Gm@len##1##2{\setlength{##1}{##2}}%
  \the\Gm@dimlist}%
```

`\Gm@setsize` The macro sets `paperwidth` and `paperheight` dimensions using `\Gm@setlength` macro.

```
\def\Gm@setsize#1(#2,#3)#4{%
  \let\Gm@td\relax
  \expandafter\Gm@setlength\csname #1width\endcsname{#2\Gm@td #4}%
  \expandafter\Gm@setlength\csname #1height\endcsname{#3\Gm@td #4}%
  \ifGm@landscape\Gm@swap@papersizetrue\else\Gm@swap@papersizefalse\fi}%
```

`\Gm@setpaper@ifpre` The macro changes the paper size.

```
\def\Gm@setpaper@ifpre#1{%
  \ifGm@preamble{#1}{\def\Gm@paper{#1}\@nameuse{Gm@#1}{paper}}}%
```

Various paper size are defined here.

```
\@namedef{Gm@a0paper}#1{\Gm@setsize{#1}(841,1189){mm}}% ISO A0
\@namedef{Gm@a1paper}#1{\Gm@setsize{#1}(594,841){mm}}% ISO A1
\@namedef{Gm@a2paper}#1{\Gm@setsize{#1}(420,594){mm}}% ISO A2
\@namedef{Gm@a3paper}#1{\Gm@setsize{#1}(297,420){mm}}% ISO A3
\@namedef{Gm@a4paper}#1{\Gm@setsize{#1}(210,297){mm}}% ISO A4
\@namedef{Gm@a5paper}#1{\Gm@setsize{#1}(148,210){mm}}% ISO A5
\@namedef{Gm@a6paper}#1{\Gm@setsize{#1}(105,148){mm}}% ISO A6
\@namedef{Gm@b0paper}#1{\Gm@setsize{#1}(1000,1414){mm}}% ISO B0
\@namedef{Gm@b1paper}#1{\Gm@setsize{#1}(707,1000){mm}}% ISO B1
\@namedef{Gm@b2paper}#1{\Gm@setsize{#1}(500,707){mm}}% ISO B2
\@namedef{Gm@b3paper}#1{\Gm@setsize{#1}(353,500){mm}}% ISO B3
\@namedef{Gm@b4paper}#1{\Gm@setsize{#1}(250,353){mm}}% ISO B4
\@namedef{Gm@b5paper}#1{\Gm@setsize{#1}(176,250){mm}}% ISO B5
\@namedef{Gm@b6paper}#1{\Gm@setsize{#1}(125,176){mm}}% ISO B6
\@namedef{Gm@c0paper}#1{\Gm@setsize{#1}(917,1297){mm}}% ISO C0
\@namedef{Gm@c1paper}#1{\Gm@setsize{#1}(648,917){mm}}% ISO C1
\@namedef{Gm@c2paper}#1{\Gm@setsize{#1}(458,648){mm}}% ISO C2
\@namedef{Gm@c3paper}#1{\Gm@setsize{#1}(324,458){mm}}% ISO C3
\@namedef{Gm@c4paper}#1{\Gm@setsize{#1}(229,324){mm}}% ISO C4
\@namedef{Gm@c5paper}#1{\Gm@setsize{#1}(162,229){mm}}% ISO C5
\@namedef{Gm@c6paper}#1{\Gm@setsize{#1}(114,162){mm}}% ISO C6
\@namedef{Gm@b0j}#1{\Gm@setsize{#1}(1030,1456){mm}}% JIS B0
\@namedef{Gm@b1j}#1{\Gm@setsize{#1}(728,1030){mm}}% JIS B1
\@namedef{Gm@b2j}#1{\Gm@setsize{#1}(515,728){mm}}% JIS B2
\@namedef{Gm@b3j}#1{\Gm@setsize{#1}(364,515){mm}}% JIS B3
\@namedef{Gm@b4j}#1{\Gm@setsize{#1}(257,364){mm}}% JIS B4
\@namedef{Gm@b5j}#1{\Gm@setsize{#1}(182,257){mm}}% JIS B5
\@namedef{Gm@b6j}#1{\Gm@setsize{#1}(128,182){mm}}% JIS B6
\@namedef{Gm@ansiapaper}#1{\Gm@setsize{#1}(8.5,11){in}}%
\@namedef{Gm@ansibpaper}#1{\Gm@setsize{#1}(11,17){in}}%
\@namedef{Gm@ansicpaper}#1{\Gm@setsize{#1}(17,22){in}}%
\@namedef{Gm@ansidpaper}#1{\Gm@setsize{#1}(22,34){in}}%
\@namedef{Gm@ansiepaper}#1{\Gm@setsize{#1}(34,44){in}}%
\@namedef{Gm@letterpaper}#1{\Gm@setsize{#1}(8.5,11){in}}%
\@namedef{Gm@legalpaper}#1{\Gm@setsize{#1}(8.5,14){in}}%
\@namedef{Gm@executivepaper}#1{\Gm@setsize{#1}(7.25,10.5){in}}%
\@namedef{Gm@screen}#1{\Gm@setsize{#1}(225,180){mm}}%
```

`'paper'` `paper` takes a paper name as its value.

```
\define@key{Gm}{paper}{\setkeys{Gm}{#1}}%
\let\KV@Gm@papername\KV@Gm@paper
```

`'a[0-6]paper'` `'b[0-6]paper'` `'b[0-6]j'` `'ansi[a-e]paper'` `'letterpaper'` `'legalpaper'` `'executivepaper'` `'screen'` The following paper names are available.

```
\define@key{Gm}{a0paper}[true]{\Gm@setpaper@ifpre{a0paper}}%
\define@key{Gm}{a1paper}[true]{\Gm@setpaper@ifpre{a1paper}}%
\define@key{Gm}{a2paper}[true]{\Gm@setpaper@ifpre{a2paper}}%
\define@key{Gm}{a3paper}[true]{\Gm@setpaper@ifpre{a3paper}}%
\define@key{Gm}{a4paper}[true]{\Gm@setpaper@ifpre{a4paper}}%
\define@key{Gm}{a5paper}[true]{\Gm@setpaper@ifpre{a5paper}}%
\define@key{Gm}{a6paper}[true]{\Gm@setpaper@ifpre{a6paper}}%
\define@key{Gm}{b0paper}[true]{\Gm@setpaper@ifpre{b0paper}}%
\define@key{Gm}{b1paper}[true]{\Gm@setpaper@ifpre{b1paper}}%
\define@key{Gm}{b2paper}[true]{\Gm@setpaper@ifpre{b2paper}}%
```

```
\define@key{Gm}{b3paper}[true]{\Gm@setpaper@ifpre{b3paper}}%
\define@key{Gm}{b4paper}[true]{\Gm@setpaper@ifpre{b4paper}}%
\define@key{Gm}{b5paper}[true]{\Gm@setpaper@ifpre{b5paper}}%
\define@key{Gm}{b6paper}[true]{\Gm@setpaper@ifpre{b6paper}}%
\define@key{Gm}{c0paper}[true]{\Gm@setpaper@ifpre{c0paper}}%
\define@key{Gm}{c1paper}[true]{\Gm@setpaper@ifpre{c1paper}}%
\define@key{Gm}{c2paper}[true]{\Gm@setpaper@ifpre{c2paper}}%
\define@key{Gm}{c3paper}[true]{\Gm@setpaper@ifpre{c3paper}}%
\define@key{Gm}{c4paper}[true]{\Gm@setpaper@ifpre{c4paper}}%
\define@key{Gm}{c5paper}[true]{\Gm@setpaper@ifpre{c5paper}}%
\define@key{Gm}{c6paper}[true]{\Gm@setpaper@ifpre{c6paper}}%
\define@key{Gm}{b0j}[true]{\Gm@setpaper@ifpre{b0j}}%
\define@key{Gm}{b1j}[true]{\Gm@setpaper@ifpre{b1j}}%
\define@key{Gm}{b2j}[true]{\Gm@setpaper@ifpre{b2j}}%
\define@key{Gm}{b3j}[true]{\Gm@setpaper@ifpre{b3j}}%
\define@key{Gm}{b4j}[true]{\Gm@setpaper@ifpre{b4j}}%
\define@key{Gm}{b5j}[true]{\Gm@setpaper@ifpre{b5j}}%
\define@key{Gm}{b6j}[true]{\Gm@setpaper@ifpre{b6j}}%
\define@key{Gm}{ansiapaper}[true]{\Gm@setpaper@ifpre{ansiapaper}}%
\define@key{Gm}{ansibpaper}[true]{\Gm@setpaper@ifpre{ansibpaper}}%
\define@key{Gm}{ansicpaper}[true]{\Gm@setpaper@ifpre{ansicpaper}}%
\define@key{Gm}{ansidpaper}[true]{\Gm@setpaper@ifpre{ansidpaper}}%
\define@key{Gm}{ansiepaper}[true]{\Gm@setpaper@ifpre{ansiepaper}}%
\define@key{Gm}{letterpaper}[true]{\Gm@setpaper@ifpre{letterpaper}}%
\define@key{Gm}{legalpaper}[true]{\Gm@setpaper@ifpre{legalpaper}}%
\define@key{Gm}{executivepaper}[true]{\Gm@setpaper@ifpre{executivepaper}}%
\define@key{Gm}{screen}[true]{\Gm@setpaper@ifpre{screen}}%
```

'paperwidth' 'paperheight' 'papersize' Direct specification for paper size is also possible.

```
\define@key{Gm}{paperwidth}{\ifGm@preamble{paperwidth}{%
  \def\Gm@paper{custom}\Gm@setlength\paperwidth{#1}}}%
\define@key{Gm}{paperheight}{\ifGm@preamble{paperheight}{%
  \def\Gm@paper{custom}\Gm@setlength\paperheight{#1}}}%
\define@key{Gm}{papersize}{\ifGm@preamble{papersize}{%
  \def\Gm@paper{custom}\Gm@branch{#1}{paperwidth}{paperheight}}}%
```

'layout' 'layoutwidth' 'layoutheight' 'layoutsize' Direct specification for layout size is also possible.

```
\define@key{Gm}{layout}{\Gm@layouttrue\@nameuse{Gm@#1}{Gm@layout}}%
\let\KV@Gm@layoutname\KV@Gm@layout
\define@key{Gm}{layoutwidth}{\Gm@layouttrue\Gm@setlength\Gm@layoutwidth{#1}}%
\define@key{Gm}{layoutheight}{\Gm@layouttrue\Gm@setlength\Gm@layoutheight{#1}}%
\define@key{Gm}{layoutsize}{\Gm@branch{#1}{layoutwidth}{layoutheight}}%
```

'landscape' 'portrait' Paper orientation setting.

```
\define@key{Gm}{landscape}[true]{\ifGm@preamble{landscape}{%
  \Gm@doifelse{landscape}{#1}%
  {\ifGm@landscape\else\Gm@landscapetrue\Gm@reverse{swap@papersize}\fi}%
  {\ifGm@landscape\Gm@landscapefalse\Gm@reverse{swap@papersize}\fi}}}%
\define@key{Gm}{portrait}[true]{\ifGm@preamble{portrait}{%
  \Gm@doifelse{portrait}{#1}%
  {\ifGm@landscape\Gm@landscapefalse\Gm@reverse{swap@papersize}\fi}%
  {\ifGm@landscape\else\Gm@landscapetrue\Gm@reverse{swap@papersize}\fi}}}%
```

'hscale' 'vscale' 'scale' These options can determine the length(s) of *total body* giving *scale(s)* against the paper size.

```
\define@key{Gm}{hscale}{\Gm@hbodytrue\edef\Gm@hscale{#1}}%
\define@key{Gm}{vscale}{\Gm@vbodytrue\edef\Gm@vscale{#1}}%
\define@key{Gm}{scale}{\Gm@branch{#1}{hscale}{vscale}}%
```

'width' 'height' 'total' 'totalwidth' 'totalheight' These options give concrete dimension(s) of *total body*. `totalwidth` and `totalheight` are aliases of `width` and `height` respectively.

```
\define@key{Gm}{width}{\Gm@hbodytrue\Gm@defbylen{width}{#1}}%
\define@key{Gm}{height}{\Gm@vbodytrue\Gm@defbylen{height}{#1}}%
\define@key{Gm}{total}{\Gm@branch{#1}{width}{height}}%
\let\KV@Gm@totalwidth\KV@Gm@width
\let\KV@Gm@totalheight\KV@Gm@height
```

'textwidth' These options directly sets the dimensions \textwidth and \textheight. body is an alias of text.
'textheight'
'text'
'body'

```
\define@key{Gm}{textwidth}{\Gm@hbodytrue\Gm@defbylen{textwidth}{#1}}%
\define@key{Gm}{textheight}{\Gm@vbodytrue\Gm@defbylen{textheight}{#1}}%
\define@key{Gm}{text}{\Gm@branch{#1}{textwidth}{textheight}}%
\let\KV@Gm@body\KV@Gm@text
```

'lines' The option sets \textheight with the number of lines.

```
\define@key{Gm}{lines}{\Gm@vbodytrue\Gm@defbycnt{lines}{#1}}%
```

'includehead' The options take the corresponding dimensions as part of *body*.
'includefoot'
'includeheadfoot'
'includemp'
'includeall'

```
\define@key{Gm}{includehead}[true]{\Gm@setbool{includehead}{#1}}%
\define@key{Gm}{includefoot}[true]{\Gm@setbool{includefoot}{#1}}%
\define@key{Gm}{includeheadfoot}[true]{\Gm@doifelse{includeheadfoot}{#1}%
  {\Gm@includeheadtrue\Gm@includefoottrue}%
  {\Gm@includeheadfalse\Gm@includefootfalse}}%
\define@key{Gm}{includemp}[true]{\Gm@setbool{includemp}{#1}}%
\define@key{Gm}{includeall}[true]{\Gm@doifelse{includeall}{#1}%
  {\Gm@includeheadtrue\Gm@includefoottrue\Gm@includemptrue}%
  {\Gm@includeheadfalse\Gm@includefootfalse\Gm@includempfalse}}%
```

'ignorehead' These options exclude *head*, *foot* and *marginpars* when determining *body*.
'ignorefoot'
'ignoreheadfoot'
'ignoremp'
'ignoreall'

```
\define@key{Gm}{ignorehead}[true]{%
  \Gm@setboolrev[ignorehead]{includehead}{#1}}%
\define@key{Gm}{ignorefoot}[true]{%
  \Gm@setboolrev[ignorefoot]{includefoot}{#1}}%
\define@key{Gm}{ignoreheadfoot}[true]{\Gm@doifelse{ignoreheadfoot}{#1}%
  {\Gm@includeheadfalse\Gm@includefootfalse}%
  {\Gm@includeheadtrue\Gm@includefoottrue}}%
\define@key{Gm}{ignoremp}[true]{%
  \Gm@setboolrev[ignoremp]{includemp}{#1}}%
\define@key{Gm}{ignoreall}[true]{\Gm@doifelse{ignoreall}{#1}%
  {\Gm@includeheadfalse\Gm@includefootfalse\Gm@includempfalse}%
  {\Gm@includeheadtrue\Gm@includefoottrue\Gm@includemptrue}}%
```

'heightrounded' The option rounds \textheight to n-times of \baselineskip plus \topskip.

```
\define@key{Gm}{heightrounded}[true]{\Gm@setbool{heightrounded}{#1}}%
```

'hdivide' The options are useful to specify partitioning in each direction of the paper.
'vdivide'
'divide'

```
\define@key{Gm}{hdivide}{\Gm@parse@divide{#1}{lmargin}{width}{rmargin}}%
\define@key{Gm}{vdivide}{\Gm@parse@divide{#1}{tmargin}{height}{bmargin}}%
\define@key{Gm}{divide}{\Gm@parse@divide{#1}{lmargin}{width}{rmargin}%
  \Gm@parse@divide{#1}{tmargin}{height}{bmargin}}%
```

'lmargin' These options set *margins*. left, inner, innermargin are aliases of lmargin. right, outer, outermargin are aliases of rmargin. top and bottom are aliases of tmargin and bmargin respectively.
'rmargin'
'tmargin'
'bmargin'
'left'
'inner'
'innermargin'
'right'
'outer'
'outermargin'
'top'
'bottom'

```
\define@key{Gm}{lmargin}{\Gm@defbylen{lmargin}{#1}}%
\define@key{Gm}{rmargin}{\Gm@defbylen{rmargin}{#1}}%
\let\KV@Gm@left\KV@Gm@lmargin
\let\KV@Gm@inner\KV@Gm@lmargin
\let\KV@Gm@innermargin\KV@Gm@lmargin
\let\KV@Gm@right\KV@Gm@rmargin
\let\KV@Gm@outer\KV@Gm@rmargin
\let\KV@Gm@outermargin\KV@Gm@rmargin
\define@key{Gm}{tmargin}{\Gm@defbylen{tmargin}{#1}}%
\define@key{Gm}{bmargin}{\Gm@defbylen{bmargin}{#1}}%
\let\KV@Gm@top\KV@Gm@tmargin
\let\KV@Gm@bottom\KV@Gm@bmargin
```

'hmargin' These options are shorthands for setting *margins*.
'vmargin'
'margin'

```
\define@key{Gm}{hmargin}{\Gm@branch{#1}{lmargin}{rmargin}}%
\define@key{Gm}{vmargin}{\Gm@branch{#1}{tmargin}{bmargin}}%
\define@key{Gm}{margin}{\Gm@branch{#1}{lmargin}{tmargin}%
  \Gm@branch{#1}{rmargin}{bmargin}}%
```

‘hmarginratio’ ‘vmarginratio’ ‘marginratio’ ‘hratio’ ‘vratio’ ‘ratio’ Options specifying the margin ratios.

```
\define@key{Gm}{hmarginratio}{\edef\Gm@hmarginratio{#1}}%
\define@key{Gm}{vmarginratio}{\edef\Gm@vmarginratio{#1}}%
\define@key{Gm}{marginratio}{\Gm@branch{#1}{hmarginratio}{vmarginratio}}%
\let\KV@Gm@hratio\KV@Gm@hmarginratio
\let\KV@Gm@vratio\KV@Gm@vmarginratio
\let\KV@Gm@ratio\KV@Gm@marginratio
```

‘hcentering’ ‘vcentering’ ‘centering’ Useful shorthands to place *body* centered.

```
\define@key{Gm}{hcentering}[true]{\Gm@doifelse{hcentering}{#1}%
  {\def\Gm@hmarginratio{1:1}}{}%
\define@key{Gm}{vcentering}[true]{\Gm@doifelse{vcentering}{#1}%
  {\def\Gm@vmarginratio{1:1}}{}%
\define@key{Gm}{centering}[true]{\Gm@doifelse{centering}{#1}%
  {\def\Gm@hmarginratio{1:1}\def\Gm@vmarginratio{1:1}}{}%
```

‘twoside’ If `twoside=true`, `\@twoside` and `\@mparswitch` is set to `true`.

```
\define@key{Gm}{twoside}[true]{\Gm@doifelse{twoside}{#1}%
  {\@twosidetrue\@mparswitchtrue}{\@twosidefalse\@mparswitchfalse}}%
```

‘asymmetric’ `asymmetric` sets `\@mparswitchfalse` and `\@twosidetrue` A `asymmetric=false` has no effect.

```
\define@key{Gm}{asymmetric}[true]{\Gm@doifelse{asymmetric}{#1}%
  {\@twosidetrue\@mparswitchfalse}{}}%
```

‘bindingoffset’ The macro adds the specified space to the inner margin.

```
\define@key{Gm}{bindingoffset}{\Gm@setlength\Gm@bindingoffset{#1}}%
```

‘headheight’ ‘headsep’ ‘footskip’ ‘head’ ‘foot’ The direct settings of *head* and/or *foot* dimensions.

```
\define@key{Gm}{headheight}{\Gm@setlength\headheight{#1}}%
\define@key{Gm}{headsep}{\Gm@setlength\headsep{#1}}%
\define@key{Gm}{footskip}{\Gm@setlength\footskip{#1}}%
\let\KV@Gm@head\KV@Gm@headheight
\let\KV@Gm@foot\KV@Gm@footskip
```

‘nohead’ ‘nofoot’ ‘noheadfoot’ They are only shorthands to set *head* and/or *foot* to be `0pt`.

```
\define@key{Gm}{nohead}[true]{\Gm@doifelse{nohead}{#1}%
  {\Gm@setlength\headheight\z@\Gm@setlength\headsep\z@}{}}%
\define@key{Gm}{nofoot}[true]{\Gm@doifelse{nofoot}{#1}%
  {\Gm@setlength\footskip\z@}{}}%
\define@key{Gm}{noheadfoot}[true]{\Gm@doifelse{noheadfoot}{#1}%
  {\Gm@setlength\headheight\z@\Gm@setlength\headsep
  \z@\Gm@setlength\footskip\z@}{}}%
```

‘footnotesep’ The option directly sets a native dimension `\footnotesep`.

```
\define@key{Gm}{footnotesep}{\Gm@setlength{\skip\footins}{#1}}%
```

‘marginparwidth’ ‘marginpar’ ‘marginparsep’ They directly set native dimensions `\marginparwidth` and `\marginparsep`.

```
\define@key{Gm}{marginparwidth}{\Gm@setlength\marginparwidth{#1}}%
\let\KV@Gm@marginpar\KV@Gm@marginparwidth
\define@key{Gm}{marginparsep}{\Gm@setlength\marginparsep{#1}}%
```

‘nomarginpar’ The macro is a shorthand for `\marginparwidth=0pt` and `\marginparsep=0pt`.

```
\define@key{Gm}{nomarginpar}[true]{\Gm@doifelse{nomarginpar}{#1}%
  {\Gm@setlength\marginparwidth\z@\Gm@setlength\marginparsep\z@}{}}%
```

‘columnsep’ The option sets a native dimension `\columnsep`.

```
\define@key{Gm}{columnsep}{\Gm@setlength\columnsep{#1}}%
```

‘hoffset’ ‘voffset’ ‘offset’ The former two options set native dimensions `\hoffset` and `\voffset`. `offset` can set both of them with the same value.

```
\define@key{Gm}{hoffset}{\Gm@setlength\hoffset{#1}}%
\define@key{Gm}{voffset}{\Gm@setlength\voffset{#1}}%
\define@key{Gm}{offset}{\Gm@branch{#1}{hoffset}{voffset}}%
```

‘layouthoffset’
‘layoutvoffset’
‘layoutoffset’
```
\define@key{Gm}{layouthoffset}{\Gm@setlength\Gm@layouthoffset{#1}}%
\define@key{Gm}{layoutvoffset}{\Gm@setlength\Gm@layoutvoffset{#1}}%
\define@key{Gm}{layoutoffset}{\Gm@branch{#1}{layouthoffset}{layoutvoffset}}%
```

‘twocolumn’ The option sets `\twocolumn` switch.

```
\define@key{Gm}{twocolumn}[true]{%
  \Gm@doif{twocolumn}{#1}{\csname @twocolumn\Gm@bool\endcsname}}%
```

‘onecolumn’ This option has the reverse effect of `twocolumn` option.

```
\define@key{Gm}{onecolumn}[true]{%
  \Gm@doifelse{onecolumn}{#1}{\@twocolumnfalse}{\@twocolumntrue}}%
```

‘reversemp’ The both options set `\reversemargin`.

‘reversemarginpar’

```
\define@key{Gm}{reversemp}[true]{%
  \Gm@doif{reversemp}{#1}{\csname @reversemargin\Gm@bool\endcsname}}%
\define@key{Gm}{reversemarginpar}[true]{%
  \Gm@doif{reversemarginpar}{#1}{\csname @reversemargin\Gm@bool\endcsname}}%
```

‘dviver’

```
\define@key{Gm}{driver}{\ifGm@preamble{driver}{%
  \edef\@@tempa{#1}\edef\@@auto{auto}\edef\@@none{none}%
  \ifx\@@tempa\@empty\let\Gm@driver\relax\else
  \ifx\@@tempa\@@none\let\Gm@driver\relax\else
  \ifx\@@tempa\@@auto\let\Gm@driver\@empty\else
  \setkeys{Gm}{#1}\fi\fi\fi\let\@@auto\relax\let\@@none\relax}}%
```

‘dvips’ ‘dvipdfm’ ‘dvipdfmx’ ‘xdvipdfmx’ ‘pdftex’ ‘luatex’ ‘xetex’ ‘vtex’ The geometry package supports `dvips`, `dvipdfm`, `pdflatex`, `luatex` and `vtex`. `dvipdfm` works like `dvips`.

```
\define@key{Gm}{dvips}[true]{\ifGm@preamble{dvips}{%
  \Gm@doifelse{dvips}{#1}{\Gm@setdriver{dvips}}{\Gm@unsetdriver{dvips}}}}%
\define@key{Gm}{dvipdfm}[true]{\ifGm@preamble{dvipdfm}{%
  \Gm@doifelse{dvipdfm}{#1}{\Gm@setdriver{dvipdfm}}{\Gm@unsetdriver{dvipdfm}}}}%
\define@key{Gm}{dvipdfmx}[true]{\ifGm@preamble{dvipdfm}{%
  \Gm@doifelse{dvipdfm}{#1}{\Gm@setdriver{dvipdfm}}{\Gm@unsetdriver{dvipdfm}}}}%
\define@key{Gm}{xdvipdfmx}[true]{\ifGm@preamble{dvipdfm}{%
  \Gm@doifelse{dvipdfm}{#1}{\Gm@setdriver{dvipdfm}}{\Gm@unsetdriver{dvipdfm}}}}%
\define@key{Gm}{pdftex}[true]{\ifGm@preamble{pdftex}{%
  \Gm@doifelse{pdftex}{#1}{\Gm@setdriver{pdftex}}{\Gm@unsetdriver{pdftex}}}}%
\define@key{Gm}{luatex}[true]{\ifGm@preamble{luatex}{%
  \Gm@doifelse{luatex}{#1}{\Gm@setdriver{luatex}}{\Gm@unsetdriver{luatex}}}}%
\define@key{Gm}{xetex}[true]{\ifGm@preamble{xetex}{%
  \Gm@doifelse{xetex}{#1}{\Gm@setdriver{xetex}}{\Gm@unsetdriver{xetex}}}}%
\define@key{Gm}{vtex}[true]{\ifGm@preamble{vtex}{%
  \Gm@doifelse{vtex}{#1}{\Gm@setdriver{vtex}}{\Gm@unsetdriver{vtex}}}}%
```

‘verbose’ The verbose mode.

```
\define@key{Gm}{verbose}[true]{\ifGm@preamble{verbose}{\Gm@setbool{verbose}{#1}}}%
```

‘reset’ The option cancels all the options specified before `reset`, except `pass`. `mag` ($\neq 1000$) with `truedimen` cannot be also reset.

```
\define@key{Gm}{reset}[true]{\ifGm@preamble{reset}{%
  \Gm@doifelse{reset}{#1}{\Gm@restore@org\Gm@initall
  \ProcessOptionsKV[c]{Gm}\Gm@setdefaultpaper}{}}}%
```

‘resetpaper’ If `resetpaper` is set to `true`, the paper size redefined in the package is discarded and the original one is restored. This option may be useful to print nonstandard sized documents with normal printers and papers.

```
\define@key{Gm}{resetpaper}[true]{\ifGm@preamble{resetpaper}{%
  \Gm@setbool{resetpaper}{#1}}}%
```

‘mag’ `mag` is expanded immediately when it is specified. So `reset` can't reset `mag` when it is set with `truedimen`.

```
\define@key{Gm}{mag}{\ifGm@preamble{mag}{\mag=#1}}%
```

‘truedimen’ If `truedimen` is set to `true`, all of the internal explicit dimensions is changed to *true* dimensions, e.g., `1in` is changed to `1truein`.

```
\define@key{Gm}{truedimen}[true]{\ifGm@preamble{truedimen}{%
  \Gm@doifelse{truedimen}{#1}{\let\Gm@truedimen\Gm@true}%
  {\let\Gm@truedimen\@empty}}}%
```

‘pass’ The option makes all the options specified ineffective except verbose switch.

```
\define@key{Gm}{pass}[true]{\ifGm@preamble{pass}{\Gm@setbool{pass}{#1}}}%
```

‘showframe’ The showframe option prints page frames to help you understand what the resulting layout is like.

```
\define@key{Gm}{showframe}[true]{\Gm@setbool{showframe}{#1}}%
```

‘showcrop’ The showcrop option prints crop marks at each corner of the layout area.

```
\define@key{Gm}{showcrop}[true]{\Gm@setbool{showcrop}{#1}}%
```

`\Gm@setdefaultpaper` The macro stores paper dimensions. This macro should be called after `\ProcessOptionsKV[c]{Gm}`. If the `landscape` option in `\documentclass` is specified, the class immediately swaps the paper dimensions.

```
\def\Gm@setdefaultpaper{%
  \ifx\Gm@paper\@undefined
    \Gm@setsize{paper}(\strip@pt\paperwidth,\strip@pt\paperheight){pt}%
    \Gm@setsize{Gm@layout}(\strip@pt\paperwidth,\strip@pt\paperheight){pt}%
    \Gm@swap@papersizefalse
  \fi}%
```

`\Gm@adjustpaper` The macro checks if paperwidth/height is larger than 0pt, which is used in `\Gm@process`. The paper dimensions can be swapped when paper orientation is changed over by `landscape` and `portrait` options.

```
\def\Gm@adjustpaper{%
  \ifdim\paperwidth>\p@\else
    \PackageError{geometry}{%
    \string\paperwidth\space(\the\paperwidth) too short}{%
    Set a paper type (e.g., ‘a4paper’).}%
  \fi
  \ifdim\paperheight>\p@\else
    \PackageError{geometry}{%
    \string\paperheight\space(\the\paperheight) too short}{%
    Set a paper type (e.g., ‘a4paper’).}%
  \fi
  \ifGm@swap@papersize
    \setlength\@tempdima{\paperwidth}%
    \setlength\paperwidth{\paperheight}%
    \setlength\paperheight{\@tempdima}%
  \fi
  \ifGm@layout\else
    \setlength\Gm@layoutwidth{\paperwidth}%
    \setlength\Gm@layoutheight{\paperheight}%
  \fi}%
```

`\Gm@checkmp` The macro checks whether or not the marginpars overrun the page.

```
\def\Gm@checkmp{%
  \ifGm@includemp\else
    \@tempcnta\z@\@tempcntb\@ne
    \if@twocolumn
      \@tempcnta\@ne
    \else
      \if@reversemargin
        \@tempcnta\@ne\@tempcntb\z@
      \fi
    \fi
    \@tempdima\marginparwidth
    \advance\@tempdima\marginparsep
    \ifnum\@tempcnta=\@ne
      \@tempdimc\@tempdima
```

```
      \setlength\@tempdimb{\Gm@lmargin}%
      \advance\@tempdimc-\@tempdimb
      \ifdim\@tempdimc>\z@
        \Gm@warning{The marginal notes overrun the paper edge.^^J
        \@spaces Add \the\@tempdimc\space and more to the left margin}%
      \fi
    \fi
    \ifnum\@tempcntb=\@ne
      \@tempdimc\@tempdima
      \setlength\@tempdimb{\Gm@rmargin}%
      \advance\@tempdimc-\@tempdimb
      \ifdim\@tempdimc>\z@
        \Gm@warning{The marginal notes overrun the paper.^^J
        \@spaces Add \the\@tempdimc\space and more to the right margin}%
      \fi
    \fi
  \fi}%
```

`\Gm@adjustmp` The macro sets marginpar correction when `includemp` is set, which is used in `\Gm@process`. The variables `\Gm@wd@mp`, `\Gm@odd@mp` and `\Gm@even@mp` are set here. Note that `\Gm@even@mp` should be used only for twoside layout.

```
\def\Gm@adjustmp{%
  \ifGm@includemp
    \@tempdimb\marginparwidth
    \advance\@tempdimb\marginparsep
    \Gm@wd@mp\@tempdimb
    \Gm@odd@mp\z@
    \Gm@even@mp\z@
    \if@twocolumn
      \Gm@wd@mp2\@tempdimb
      \Gm@odd@mp\@tempdimb
      \Gm@even@mp\@tempdimb
    \else
      \if@reversemargin
        \Gm@odd@mp\@tempdimb
        \if@mparswitch\else
          \Gm@even@mp\@tempdimb
        \fi
      \else
        \if@mparswitch
          \Gm@even@mp\@tempdimb
        \fi
      \fi
    \fi
  \fi}%
```

`\Gm@adjustbody` If the horizontal dimension of *body* is specified by user, `\Gm@width` is set properly here.

```
\def\Gm@adjustbody{
  \ifGm@hbody
    \ifx\Gm@width\@undefined
      \ifx\Gm@hscale\@undefined
        \Gm@defbylen{width}{\Gm@Dhscale\Gm@layoutwidth}%
      \else
        \Gm@defbylen{width}{\Gm@hscale\Gm@layoutwidth}%
      \fi
    \fi
    \ifx\Gm@textwidth\@undefined\else
      \setlength\@tempdima{\Gm@textwidth}%
      \ifGm@includemp
        \advance\@tempdima\Gm@wd@mp
      \fi
      \edef\Gm@width{\the\@tempdima}%
    \fi
  \fi
```

If the vertical dimension of *body* is specified by user, `\Gm@height` is set properly here.

```
\ifGm@vbody
  \ifx\Gm@height\@undefined
    \ifx\Gm@vscale\@undefined
      \Gm@defbylen{height}{\Gm@Dvscale\Gm@layoutheight}%
    \else
      \Gm@defbylen{height}{\Gm@vscale\Gm@layoutheight}%
    \fi
  \fi
  \ifx\Gm@lines\@undefined\else
```

`\topskip` has to be adjusted so that the formula "`\textheight` = (*lines* − 1) × `\baselineskip` + `\topskip`" to be correct even if large font sizes are specified by users. If `\topskip` is smaller than `\ht\strutbox`, then `\topskip` is set to `\ht\strutbox`.

```
    \ifdim\topskip<\ht\strutbox
      \setlength\@tempdima{\topskip}%
      \setlength\topskip{\ht\strutbox}%
      \Gm@warning{\noexpand\topskip was changed from \the\@tempdima\space
      to \the\topskip}%
    \fi
    \setlength\@tempdima{\baselineskip}%
    \multiply\@tempdima\Gm@lines
    \addtolength\@tempdima{\topskip}%
    \addtolength\@tempdima{-\baselineskip}%
    \edef\Gm@textheight{\the\@tempdima}%
  \fi
  \ifx\Gm@textheight\@undefined\else
    \setlength\@tempdima{\Gm@textheight}%
    \ifGm@includehead
      \addtolength\@tempdima{\headheight}%
      \addtolength\@tempdima{\headsep}%
    \fi
    \ifGm@includefoot
      \addtolength\@tempdima{\footskip}%
    \fi
    \edef\Gm@height{\the\@tempdima}%
  \fi
\fi}%
```

`\Gm@process` The main macro processing the specified dimensions is defined.

```
\def\Gm@process{%
```

If `pass` is set, the original dimensions and switches are restored and process is ended here.

```
  \ifGm@pass
    \Gm@restore@org
  \else
    \Gm@@process
  \fi}%
```

The main processing macro.

```
\def\Gm@@process{%
  \Gm@expandlengths
  \Gm@adjustpaper
  \addtolength\Gm@layoutwidth{-\Gm@bindingoffset}%
  \Gm@adjustmp
  \Gm@adjustbody
  \Gm@detall{h}{width}{lmargin}{rmargin}%
  \Gm@detall{v}{height}{tmargin}{bmargin}%
```

The real dimensions are set properly according to the result of the auto-completion calculation.

```
  \setlength\textwidth{\Gm@width}%
  \setlength\textheight{\Gm@height}%
  \setlength\topmargin{\Gm@tmargin}%
  \setlength\oddsidemargin{\Gm@lmargin}%
  \addtolength\oddsidemargin{-1\Gm@truedimen in}%
```

If `includemp` is set to `true`, `\textwidth` and `\oddsidemargin` are adjusted.

```
\ifGm@includemp
  \advance\textwidth-\Gm@wd@mp
  \advance\oddsidemargin\Gm@odd@mp
\fi
```

Determining `\evensidemargin`. In the twoside page layout, the right margin value `\Gm@rmargin` is used. If the marginal note width is included, `\evensidemargin` should be corrected by `\Gm@even@mp`.

```
\if@mparswitch
  \setlength\evensidemargin{\Gm@rmargin}%
  \addtolength\evensidemargin{-1\Gm@truedimen in}%
  \ifGm@includemp
    \advance\evensidemargin\Gm@even@mp
  \fi
\else
  \evensidemargin\oddsidemargin
\fi
```

The bindingoffset correction for `\oddsidemargin`.

```
\advance\oddsidemargin\Gm@bindingoffset
\addtolength\topmargin{-1\Gm@truedimen in}%
```

If the head of the page is included in *total body*, `\headheight` and `\headsep` are removed from `\textheight`, otherwise from `\topmargin`.

```
\ifGm@includehead
  \addtolength\textheight{-\headheight}%
  \addtolength\textheight{-\headsep}%
\else
  \addtolength\topmargin{-\headheight}%
  \addtolength\topmargin{-\headsep}%
\fi
```

If the foot of the page is included in *total body*, `\footskip` is removed from `\textheight`.

```
\ifGm@includefoot
  \addtolength\textheight{-\footskip}%
\fi
```

If `heightrounded` is set, `\textheight` is rounded.

```
\ifGm@heightrounded
  \setlength\@tempdima{\textheight}%
  \addtolength\@tempdima{-\topskip}%
  \@tempcnta\@tempdima
  \@tempcntb\baselineskip
  \divide\@tempcnta\@tempcntb
  \setlength\@tempdimb{\baselineskip}%
  \multiply\@tempdimb\@tempcnta
  \advance\@tempdima-\@tempdimb
  \multiply\@tempdima\tw@
  \ifdim\@tempdima>\baselineskip
    \addtolength\@tempdimb{\baselineskip}%
  \fi
  \addtolength\@tempdimb{\topskip}%
  \textheight\@tempdimb
\fi
```

The paper width is set back by adding `\Gm@bindingoffset`.

```
\advance\oddsidemargin\Gm@layouthoffset%
\advance\evensidemargin\Gm@layouthoffset%
\advance\topmargin\Gm@layoutvoffset%
\addtolength\Gm@layoutwidth{\Gm@bindingoffset}%
}% end of \Gm@@process
```

`\Gm@detectdriver` The macro checks the typeset environment and changes the driver option if necessary. To make the engine detection more robust, the macro is rewritten with packages ifpdf, ifvtex and ifxetex.

```
\def\Gm@detectdriver{%
```

If the driver option is not specified explicitly, then driver auto-detection works.

```
\ifx\Gm@driver\@empty
  \typeout{*geometry* driver: auto-detecting}%
```

`\ifpdf` is defined in ifpdf package in 'oberdiek' bundle.

```
  \ifpdf
    \ifx\pdfextension\@undefined
      \Gm@setdriver{pdftex}%
    \else
      \Gm@setdriver{luatex}%
    \fi
  \else
    \Gm@setdriver{dvips}%
  \fi
```

`\ifvtex` is defined in ifvtex package in 'oberdiek' bundle.

```
  \ifvtex
    \Gm@setdriver{vtex}%
  \fi
```

`\ifxetex` is defined in ifxetex package written by Will Robertson.

```
  \ifxetex
    \Gm@setdriver{xetex}
  \fi
```

When the driver option is set by the user, check if it is valid or not.

```
\else
  \ifx\Gm@driver\Gm@xetex %%
    \ifxetex\else
      \Gm@warning{Wrong driver setting: 'xetex'; trying 'pdftex' driver}%
      \Gm@setdriver{pdftex}
    \fi
  \fi
  \ifx\Gm@driver\Gm@vtex
    \ifvtex\else
      \Gm@warning{Wrong driver setting: 'vtex'; trying 'dvips' driver}%
      \Gm@setdriver{dvips}%
    \fi
  \fi
\fi
\ifx\Gm@driver\relax
  \typeout{*geometry* detected driver: <none>}%
\else
  \typeout{*geometry* detected driver: \Gm@driver}%
\fi}%
```

`\Gm@showparams` Prints the resulted parammeters and dimensions to STDOUT if `verbose` is true. `\Gm@width` and `\Gm@height` are expanded to get the real size.

```
\def\Gm@showparams#1{%
  \ifGm@verbose\expandafter\typeout\else\expandafter\wlog\fi
  {\Gm@logcontent{#1}}}%
\def\Gm@showdim#1{* \string#1=\the#1^^J}%
\def\Gm@showbool#1{\@nameuse{ifGm@#1}#1\space\fi}%
```

`\Gm@logcontent` The content of geometry paramenters and native dimensions for the page layout.

```
\def\Gm@logcontent#1{%
  *geometry* verbose mode - [ #1 ] result:^^J%
  \ifGm@pass * pass: disregarded the geometry package!^^J%
  \else
  * driver: \if\Gm@driver<none>\else\Gm@driver\fi^^J%
  * paper: \ifx\Gm@paper\@undefined<default>\else\Gm@paper\fi^^J%
  * layout: \ifGm@layout<custom>\else<same size as paper>\fi^^J%
  \ifGm@layout
  * layout(width,height): (\the\Gm@layoutwidth,\the\Gm@layoutheight)^^J%
  \fi
  * layoutoffset:(h,v)=(\the\Gm@layouthoffset,\the\Gm@layoutvoffset)^^J%
```

```
  \@ifundefined{Gm@lines}{}{* lines: \Gm@lines^^J}%
  \@ifundefined{Gm@hmarginratio}{}{* hratio: \Gm@hmarginratio^^J}%
  \@ifundefined{Gm@vmarginratio}{}{* vratio: \Gm@vmarginratio^^J}%
  \ifdim\Gm@bindingoffset=\z@\else
  * bindingoffset: \the\Gm@bindingoffset^^J\fi
  * modes: %
   \Gm@showbool{landscape}%
   \Gm@showbool{includehead}%
   \Gm@showbool{includefoot}%
   \Gm@showbool{includemp}%
   \if@twoside twoside\space\fi%
   \if@mparswitch\else\if@twoside asymmetric\space\fi\fi%
   \Gm@showbool{heightrounded}%
   \ifx\Gm@truedimen\@empty\else truedimen\space\fi%
   \Gm@showbool{showframe}%
   \Gm@showbool{showcrop}%
  ^^J%
  * h-part:(L,W,R)=(\Gm@lmargin, \Gm@width, \Gm@rmargin)^^J%
  * v-part:(T,H,B)=(\Gm@tmargin, \Gm@height, \Gm@bmargin)^^J%
  \fi
  \Gm@showdim{\paperwidth}%
  \Gm@showdim{\paperheight}%
  \Gm@showdim{\textwidth}%
  \Gm@showdim{\textheight}%
  \Gm@showdim{\oddsidemargin}%
  \Gm@showdim{\evensidemargin}%
  \Gm@showdim{\topmargin}%
  \Gm@showdim{\headheight}%
  \Gm@showdim{\headsep}%
  \Gm@showdim{\topskip}%
  \Gm@showdim{\footskip}%
  \Gm@showdim{\marginparwidth}%
  \Gm@showdim{\marginparsep}%
  \Gm@showdim{\columnsep}%
  * \string\skip\string\footins=\the\skip\footins^^J%
  \Gm@showdim{\hoffset}%
  \Gm@showdim{\voffset}%
  \Gm@showdim{\mag}%
  * \string\@twocolumn\if@twocolumn true\else false\fi^^J%
  * \string\@twoside\if@twoside true\else false\fi^^J%
  * \string\@mparswitch\if@mparswitch true\else false\fi^^J%
  * \string\@reversemargin\if@reversemargin true\else false\fi^^J%
  * (1in=72.27pt=25.4mm, 1cm=28.453pt)^^J}%
```

Macros for the page frames and cropmarks.

```
\def\Gm@cropmark(#1,#2,#3,#4){%
  \begin{picture}(0,0)
    \setlength\unitlength{1truemm}%
    \linethickness{0.25pt}%
    \put(#3,0){\line(#1,0){17}}%
    \put(0,#4){\line(0,#2){17}}%
  \end{picture}}%
\providecommand*\vb@xt@{\vbox to}%
\def\Gm@vrule{\vrule width 0.2pt height\textheight depth\z@}%
\def\Gm@hrule{\hrule height 0.2pt depth\z@ width\textwidth}%
\def\Gm@hruled{\hrule height\z@ depth0.2pt width\textwidth}%
\newcommand*{\Gm@vrules@mpi}{%
  \hb@xt@\@tempdima{\llap{\Gm@vrule}\ignorespaces
  \hskip \textwidth\Gm@vrule\hskip \marginparsep
  \llap{\Gm@vrule}\hfil\Gm@vrule}}%
\newcommand*{\Gm@vrules@mpii}{%
  \hb@xt@\@tempdima{\hskip-\marginparwidth\hskip-\marginparsep
  \llap{\Gm@vrule}\ignorespaces
  \hskip \marginparwidth\rlap{\Gm@vrule}\hskip \marginparsep
  \llap{\Gm@vrule}\hskip\textwidth\rlap{\Gm@vrule}\hss}}%
```

```
\newcommand*{\Gm@pageframes}{%
  \vb@xt@\z@{%
   \ifGm@showcrop
    \vb@xt@\z@{\vskip-1\Gm@truedimen in\vskip\Gm@layoutvoffset%
     \hb@xt@\z@{\hskip-1\Gm@truedimen in\hskip\Gm@layouthoffset%
      \vb@xt@\Gm@layoutheight{%
       \let\protect\relax
       \hb@xt@\Gm@layoutwidth{\Gm@cropmark(-1,1,-3,3)\hfil\Gm@cropmark(1,1,3,3)}%
       \vfil
       \hb@xt@\Gm@layoutwidth{\Gm@cropmark(-1,-1,-3,-3)\hfil\Gm@cropmark(1,-1,3,-3)}}%
     \hss}%
    \vss}%
   \fi%
   \ifGm@showframe
    \if@twoside
     \ifodd\count\z@
       \let\@themargin\oddsidemargin
     \else
       \let\@themargin\evensidemargin
     \fi
    \fi
    \moveright\@themargin%
    \vb@xt@\z@{%
     \vskip\topmargin\vb@xt@\z@{\vss\Gm@hrule}%
     \vskip\headheight\vb@xt@\z@{\vss\Gm@hruled}%
     \vskip\headsep\vb@xt@\z@{\vss\Gm@hrule}%
     \@tempdima\textwidth
     \advance\@tempdima by \marginparsep
     \advance\@tempdima by \marginparwidth
     \if@mparswitch
      \ifodd\count\z@
       \Gm@vrules@mpi
      \else
       \Gm@vrules@mpii
      \fi
     \else
      \Gm@vrules@mpi
     \fi
     \vb@xt@\z@{\vss\Gm@hrule}%
     \vskip\footskip\vb@xt@\z@{\vss\Gm@hruled}%
     \vss}%
    \fi%
  }}%
```

`\ProcessOptionsKV` This macro can process class and package options using 'key=value' scheme. Only class options are processed with an optional argument '`c`', package options with '`p`' , and both of them by default.

```
\def\ProcessOptionsKV{\@ifnextchar[%]
  {\@ProcessOptionsKV}{\@ProcessOptionsKV[]}}%
\def\@ProcessOptionsKV[#1]#2{%
  \let\@tempa\@empty
  \@tempcnta\z@
  \if#1p\@tempcnta\@ne\else\if#1c\@tempcnta\tw@\fi\fi
  \ifodd\@tempcnta
   \edef\@tempa{\@ptionlist{\@currname.\@currext}}%
  \else
    \@for\CurrentOption:=\@classoptionslist\do{%
      \@ifundefined{KV@#2@\CurrentOption}%
      {}{\edef\@tempa{\@tempa,\CurrentOption,}}}%
    \ifnum\@tempcnta=\z@
      \edef\@tempa{\@tempa,\@ptionlist{\@currname.\@currext}}%
    \fi
  \fi
  \edef\@tempa{\noexpand\setkeys{#2}{\@tempa}}%
  \@tempa
  \AtEndOfPackage{\let\@unprocessedoptions\relax}}%
```

```
\def\Gm@setkeys{\setkeys{Gm}}%
```

\Gm@processconf `\ExecuteOptions` is replaced with `\Gm@setkey` to make it possible to deal with '⟨*key*⟩=⟨*value*⟩' as its argument.

```
\def\Gm@processconfig{%
  \let\Gm@origExecuteOptions\ExecuteOptions
  \let\ExecuteOptions\Gm@setkeys
  \InputIfFileExists{geometry.cfg}{}{}
  \let\ExecuteOptions\Gm@origExecuteOptions}%
```

The original page layout before loading geometry is saved here. `\Gm@restore@org` is defined here for `reset` option.

```
\Gm@save
\edef\Gm@restore@org{\Gm@restore}%
\Gm@initall
```

Processing config file.

```
\Gm@processconfig
```

The optional arguments to `\documentclass` are processed here.

```
\ProcessOptionsKV[c]{Gm}%
```

Paper dimensions given by class default are stored.

```
\Gm@setdefaultpaper
```

The optional arguments to `\usepackage` are processed here.

```
\ProcessOptionsKV[p]{Gm}%
```

Actual settings and calculation for layout dimensions are processed.

```
\Gm@process
```

\AtBeginDocument The processes for `verbose`, `showframe` and drivers are added to `\AtBeginDocument`. `\Gm@restore@org` is redefined here with the paper size specified in the preamble for `\newgeometry` to use it. This should be done before magnifying the paper size with `\mag` because the layout calculation would be affected by changing the paper size.

```
\AtBeginDocument{%
  \Gm@savelength{paperwidth}%
  \Gm@savelength{paperheight}%
  \edef\Gm@restore@org{\Gm@restore}%
```

The original paper size is used if `resetpaper`.

```
  \ifGm@resetpaper
    \edef\Gm@pw{\Gm@orgpw}%
    \edef\Gm@ph{\Gm@orgph}%
  \else
    \edef\Gm@pw{\the\paperwidth}%
    \edef\Gm@ph{\the\paperheight}%
  \fi
```

If `pass` is not set, the paper size is multiplied according to the specified `mag`.

```
  \ifGm@pass\else
    \ifnum\mag=\@m\else
      \Gm@magtooffset
      \divide\paperwidth\@m
      \multiply\paperwidth\the\mag
      \divide\paperheight\@m
      \multiply\paperheight\the\mag
    \fi
  \fi
```

Checking the driver options.

```
  \Gm@detectdriver
```

If `xetex` and `\pdfpagewidth` is defined, `\pdfpagewidth` and `\pdfpageheight` would be set.

```
  \ifx\Gm@driver\Gm@xetex
    \@ifundefined{pdfpagewidth}{}{%
      \setlength\pdfpagewidth{\Gm@pw}%
      \setlength\pdfpageheight{\Gm@ph}}%
```

```
    \ifnum\mag=\@m\else
      \ifx\Gm@truedimen\Gm@true
        \setlength\paperwidth{\Gm@pw}%
        \setlength\paperheight{\Gm@ph}%
      \fi
    \fi
  \fi
```

If `pdftex` is set to `true`, pdf-commands are set properly. To avoid `pdftex` magnification problem, `\pdfhorigin` and `\pdfvorigin` are adjusted for `\mag`.

```
  \ifx\Gm@driver\Gm@pdftex
    \@ifundefined{pdfpagewidth}{}{%
      \setlength\pdfpagewidth{\Gm@pw}%
      \setlength\pdfpageheight{\Gm@ph}}%
    \ifnum\mag=\@m\else
      \@tempdima=\mag sp%
      \@ifundefined{pdfhorigin}{}{%
        \divide\pdfhorigin\@tempdima
        \multiply\pdfhorigin\@m
        \divide\pdfvorigin\@tempdima
        \multiply\pdfvorigin\@m}%
      \ifx\Gm@truedimen\Gm@true
        \setlength\paperwidth{\Gm@pw}%
        \setlength\paperheight{\Gm@ph}%
      \fi
    \fi
  \fi
```

If `luatex` is set to `true`, pdf-commands are set properly. To avoid `luatex` magnification problem, `\horigin` and `\vorigin` are adjusted for `\mag`.

```
  \ifx\Gm@driver\Gm@luatex
    \setlength\pagewidth{\Gm@pw}%
    \setlength\pageheight{\Gm@ph}%
    \ifnum\mag=\@m\else
      \@tempdima=\mag sp
        \edef\Gm@horigin{\pdfvariable horigin}%
        \edef\Gm@vorigin{\pdfvariable vorigin}%
        \divide\Gm@horigin\@tempdima
        \multiply\Gm@horigin\@m
        \divide\Gm@vorigin\@tempdima
        \multiply\Gm@vorigin\@m
      \ifx\Gm@truedimen\Gm@true
        \setlength\paperwidth{\Gm@pw}%
        \setlength\paperheight{\Gm@ph}%
      \fi
    \fi
  \fi
```

With VTEX environment, VTEX variables are set here.

```
  \ifx\Gm@driver\Gm@vtex
    \@ifundefined{mediawidth}{}{%
      \mediawidth=\paperwidth
      \mediaheight=\paperheight}%
    \ifvtexdvi
      \AtBeginDvi{\special{papersize=\the\paperwidth,\the\paperheight}}%
    \fi
  \fi
```

If `dvips` or `dvipdfm` is specified, paper size is embedded in dvi file with `\special`. For dvips, a landscape correction is added because a landscape document converted by dvips is upside-down in PostScript viewers.

```
  \ifx\Gm@driver\Gm@dvips
    \AtBeginDvi{\special{papersize=\the\paperwidth,\the\paperheight}}%
    \ifx\Gm@driver\Gm@dvips\ifGm@landscape
      \AtBeginDvi{\special{! /landplus90 true store}}%
    \fi\fi
```

If dvipdfm is specified and atbegshi package in 'oberdiek' bundle is loaded, \AtBeginShipoutFirst is used instead of \AtBeginDvi for compatibility with hyperref and dvipdfm program.

```
\else\ifx\Gm@driver\Gm@dvipdfm
  \ifcase\ifx\AtBeginShipoutFirst\relax\@ne\else
      \ifx\AtBeginShipoutFirst\@undefined\@ne\else\z@\fi\fi
    \AtBeginShipoutFirst{\special{papersize=\the\paperwidth,\the\paperheight}}%
  \or
    \AtBeginDvi{\special{papersize=\the\paperwidth,\the\paperheight}}%
  \fi
\fi\fi
```

Page frames are shipped out when showframe=true, cropmarks for showcrop=true on each page. The atbegshi package is used for overloading \shipout.

```
\@tempswafalse
\ifGm@showframe
  \@tempswatrue
\else\ifGm@showcrop
  \@tempswatrue
\fi\fi
\if@tempswa
  \RequirePackage{atbegshi}%
    \AtBeginShipout{\setbox\AtBeginShipoutBox=\vbox{%
      \baselineskip\z@skip\lineskip\z@skip\lineskiplimit\z@
      \Gm@pageframes\box\AtBeginShipoutBox}}%
\fi
```

The layout dimensions for \restoregeometry are saved at the end of the \AtBeginDocument.

```
\Gm@save
\edef\Gm@restore@pkg{\Gm@restore}%
```

The package checks whether or not the marginpars overrun the page, if verbose and unless pass.

```
\ifGm@verbose\ifGm@pass\else\Gm@checkmp\fi\fi
```

\Gm@showparams puts the resulting parameters and dimensions into the log file. With verbose, they are shown on the terminal as well.

```
\Gm@showparams{preamble}%
```

The following lines free the memories no longer needed.

```
\let\Gm@pw\relax
\let\Gm@ph\relax
}% end of \AtBeginDocument
```

\geometry The macro \geometry can be called multiple times in the preamble (before \begin{document}).

```
\newcommand{\geometry}[1]{%
  \Gm@clean
  \setkeys{Gm}{#1}%
  \Gm@process}%
\@onlypreamble\geometry
```

\Gm@changelayout The macro, which can be called from \newgeometry, \restoregeometry and \loadgeometry, changes the layout in the middle of the document.

```
\DeclareRobustCommand\Gm@changelayout{%
  \setlength{\@colht}{\textheight}
  \setlength{\@colroom}{\textheight}%
  \setlength{\vsize}{\textheight}
  \setlength{\columnwidth}{\textwidth}%
  \if@twocolumn%
    \advance\columnwidth-\columnsep
    \divide\columnwidth\tw@%
    \@firstcolumntrue%
  \fi%
  \setlength{\hsize}{\columnwidth}%
  \setlength{\linewidth}{\hsize}}%
```

\newgeometry The macro \newgeometry, which changes the layout, can be used only in the document. It would reset the options specified in the preamble except for paper size options and \mag.

```
\newcommand{\newgeometry}[1]{%
  \clearpage
  \Gm@restore@org
  \Gm@initnewgm
  \Gm@newgmtrue
  \setkeys{Gm}{#1}%
  \Gm@newgmfalse
  \Gm@process
  \ifnum\mag=\@m\else\Gm@magtooffset\fi
  \Gm@changelayout
  \Gm@showparams{newgeometry}}%
```

`\restoregeometry` The macro restores the resulting layout specified in the preamble, namely the first-page layout right after `\begin{document}`.

```
\newcommand{\restoregeometry}{%
  \clearpage
  \Gm@restore@pkg
  \Gm@changelayout}%
```

`\savegeometry` The macro saves the layout with the name specified with the argument. The saved layout can be loaded with `\loadgeometry{`⟨*name*⟩`}`.

```
\newcommand*{\savegeometry}[1]{%
  \Gm@save
  \expandafter\edef\csname Gm@restore@@#1\endcsname{\Gm@restore}}%
```

`\loadgeometry` The macro loads the layout saved with `\savegeometry{`⟨*name*⟩`}`. If the name is not found, the macro would warn it and do nothing for the layout.

```
\newcommand*{\loadgeometry}[1]{%
  \clearpage
  \@ifundefined{Gm@restore@@#1}{%
    \PackageError{geometry}{%
    \string\loadgeometry : name `#1' undefined}{%
    The name `#1' should be predefined with \string\savegeometry}%
  }{\@nameuse{Gm@restore@@#1}%
  \Gm@changelayout}}%
⟨/package⟩
```

12 Config file

In the configuration file `geometry.cfg`, one can use `\ExecuteOptions` to set the site or user default settings.

```
⟨*config⟩
%<<SAVE_INTACT

%  Uncomment and edit the line below to set default options.
%\ExecuteOptions{a4paper}

%SAVE_INTACT
⟨/config⟩
```

13 Sample file

Here is a sample document for the geometry package.

```
⟨*samples⟩
%<<SAVE_INTACT
\documentclass[12pt]{article}% uses letterpaper by default
% \documentclass[12pt,a4paper]{article}% for A4 paper
%---------------------------------------------------------------
% Edit and uncomment one of the settings below
%---------------------------------------------------------------
% \usepackage{geometry}
% \usepackage[centering]{geometry}
```

```
% \usepackage[width=10cm,vscale=.7]{geometry}
% \usepackage[margin=1cm, papersize={12cm,19cm}, resetpaper]{geometry}
% \usepackage[margin=1cm,includeheadfoot]{geometry}
\usepackage[margin=1cm,includeheadfoot,includemp]{geometry}
% \usepackage[margin=1cm,bindingoffset=1cm,twoside]{geometry}
% \usepackage[hmarginratio=2:1, vmargin=2cm]{geometry}
% \usepackage[hscale=0.5,twoside]{geometry}
% \usepackage[hscale=0.5,asymmetric]{geometry}
% \usepackage[hscale=0.5,heightrounded]{geometry}
% \usepackage[left=1cm,right=4cm,top=2cm,includefoot]{geometry}
% \usepackage[lines=20,left=2cm,right=6cm,top=2cm,twoside]{geometry}
% \usepackage[width=15cm, marginparwidth=3cm, includemp]{geometry}
% \usepackage[hdivide={1cm,,2cm}, vdivide={3cm,8in,}, nohead]{geometry}
% \usepackage[headsep=20pt, head=40pt,foot=20pt,includeheadfoot]{geometry}
% \usepackage[text={6in,8in}, top=2cm, left=2cm]{geometry}
% \usepackage[centering,includemp,twoside,landscape]{geometry}
% \usepackage[mag=1414,margin=2cm]{geometry}
% \usepackage[mag=1414,margin=2truecm,truedimen]{geometry}
% \usepackage[a5paper, landscape, twocolumn, twoside,
%    left=2cm, hmarginratio=2:1, includemp, marginparwidth=43pt,
%    bottom=1cm, foot=.7cm, includefoot, textheight=11cm, heightrounded,
%    columnsep=1cm,verbose]{geometry}
%---------------------------------------------------------------
% No need to change below
%---------------------------------------------------------------
\geometry{verbose,showframe}% the options appended.
\usepackage{lipsum}% for dummy text of 150 paragraphs
\newcommand\mynote{\marginpar[\raggedright
A sample margin note in the left side.]%
{\raggedright A sample margin note.}}%
\newcommand\myfootnote{\footnote{This is a sample footnote text.}}
\begin{document}
\lipsum[1-2]\mynote\lipsum[3-4]\mynote
\lipsum[5-11]\mynote\lipsum[12]\myfootnote
\lipsum[13-22]\mynote\lipsum[23-32]
\end{document}
%SAVE_INTACT
```
⟨/samples⟩