latex-papersize

Calculate LaTeX paper and margin settings for arbitrary magnification

Silas S. Brown

2005–2009, 2016, 2019–20, 2025–26.
Version 1.71

License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Introduction

When producing enlarged material in LaTeX for people with low vision, it's often not enough to simply add

such things as `\Large` because that doesn't enlarge *everything* and it can be difficult to achieve the exact desired size, especially if unusual packages are being used as well.

It's more effective to change the LaTeX paper size and margin settings to simulate *small paper* and then magnify the result up to the desired physical paper size. This magnifies everything, and also adds some clarity for low-vision users because fonts like Computer Modern Roman have different versions at different sizes and the small-sized versions are often meant to be clearer.

latex-papersize calculates the necessary settings for arbitrary font and page sizes. There are two ways to run it: a

modern `.sty` that works in `pdflatex`, `xelatex` and `lualatex`, and a legacy Python script that works with `pdflatex` or plain old `latex` / `dvips`. The legacy Python script requires Python (works in both Python 2 and Python 3) but the `.sty` file can be used as-is.

Modern usage

In the following instructions, base-size is the point size that the TEX file is based on—quick reference: if the `documentclass` parameters say `12pt` then (within 1%) `\small`=11 `\normalsize`=12 `\large`=14 `\Large`=17 `\LARGE`=20 `\huge`=25—and desired-size

is the point size that you want to produce.

For the modern `.sty` file, simply use

```
\usepackage [basesize=12,
desiredsize=26] {latex-papersize}
```

(The default is to enlarge from 10 point to 26 point, but you should ideally check with individual readers which point size they work with best.)

Other parameters are the dimensions `paperwidth` and `paperheight` (defaults to A4), `marginleft` and `margintop` (defaults to 10mm), and `extrabottom` for how much space to leave for page numbers at the bottom of each page: try `extrabottom=15pt` for plain pagestyle. If it is left at its default of 0, `\pagestyle{empty}` will be set

automatically.

Legacy usage

If you need to use the legacy Python script, its command-line parameters depend on whether you are using `latex`/`dvips` or `pdflatex`. On most systems, you can begin by typing `python latex-papersize.py`. On some systems you will need to type `python3` instead of `python`, and/or you may need to specify the full path of `latex-papersize.py` depending on your distribution. If you've installed it using `pip`, you can try `python -m latex-papersize` or `python3 -m`

`latex-papersize`. You may also be able to copy `latex-papersize.py` to a directory on your PATH such as `/usr/local/bin` and then just type `latex-papersize`.

The instructions below assume you will be typing

```
python latex-papersize.py
```

but you should change this to suit your system as described in the above paragraph.

Legacy usage with `latex` and `dvips`

To print the geometry settings:

```
python latex-papersize.py base-
```

size desired-size `tex`

e.g.: `python latex-papersize.py 12 26 tex`

The output of this command should then be placed after `\documentclass` in your `.tex` file.

Then run `latex` to make a DVI file, and then do:

`python latex-papersize.py` base-size desired-size dvi-file

e.g.: `python latex-papersize.py 12 26 myfile.dvi`

which will print the appropriate `dvips` command.

Legacy usage with `pdflatex`

It should no longer be necessary to use the Python script with `pdflatex`, since the modern `.sty` file is compatible with `pdflatex`. But if for some reason you need to use the Python script instead, you need only one command:

`python latex-papersize.py` base-size desired-size `pdftex`

the output of this command should go just before `\begin{document}`. NB if using `hyperref` package, put `hyperref` *after* these settings (and if also using `pinyin` package then put `pinyin` package after that again)

Page numbers in legacy usage

When invoking latex-papersize.py with the `tex` or `pdftex` options, you can optionally add a fourth parameter to specify how many points to leave for page numbers at the bottom of each page. Normally 15 is a good idea. Leaving this out causes no room to be left for page numbers and `\pagestyle{empty}` to be added. This does not affect the `dvips` command.

Paper sizes and margins in legacy usage

It is assumed that the final physical printout will be on A4 portrait with 10mm margins. You can override this by setting the `paper_width` and `paper_height` environment variables (in millimetres) before running, e.g.: `paper_width=297 paper_height=210 python latex-papersize.py` ... You can also set the environment variables margin_left and margin_top (the right and bottom margins are assumed to be mirrors of these).

If using with non-PDF latex, remember to set the environment variables again when asking latex-papersize.py for the

dvips command. Or "export" them so they remain set.

Poster printing

If you want to print on a physical paper size that is larger than your printer can handle, you can make up the larger size by sticking together smaller pieces of paper. Perhaps the best way to do this is to use a separate utility that knows about printable areas, cut margins, etc, such as Jos van Eijndhoven's `poster` utility at https://ctan.org/pkg/poster

In this case you can give latex-papersize.py a `margin_left` and `margin_top` of 0 (because `poster` will

handle the margins), set the paper size and desired point size appropriate for the final poster, use `latex`/`dvips`, and run the resulting `.ps` through `poster` with `-s 1`. This is better than getting `poster` to scale, because if any of your fonts are rendered into bitmaps by Metafont (as some CJK fonts are) then you will likely get a better resolution if the final size is given to latex-papersize.py rather than to `poster`.

If you're aiming for a specific number of sheets of paper, don't pick a physical size that's an exact multiple of your printer's paper size, because `poster` accounts for unprintable areas and overlaps slightly. Multiply the printer paper's height and width by 0.88 (or 0.86 if you

want a visual margin added to the finished poster besides the cutting margins) and try a multiple of that size.

Example legacy usage in a script

To typeset a LaTeX file `file.tex` and magnify from 12-point to 26-point, type a line such as the following (after adjusting the documentclass it specifies, and removing or commenting out the documentclass in `file.tex`):

```
   latex "\\documentclass [12pt]
{article}
   $(python latex-papersize.py 12
26 tex) \\input {file.tex}" && mv
```

```
article.dvi file.dvi
```

Or in `pdflatex` (slightly more complex because we need to put the settings just before `\begin{document}`): `cat file.tex | awk - "/^ *\\\\\begin *\\{document\\}[^#-~]*\$/ { print \"$(python latex-papersize.py 12 26 pdftex | sed -e 's/\\/\\\\\/g')\" } { print }" > /tmp/tmp.tex` then `pdflatex "\documentclass [12pt] {article} \\input {/tmp/tmp.tex}" && mv article.pdf file.pdf`

To run `dvips` on the `.dvi` file (not needed for `pdflatex`):

```
$(python latex-papersize.py 12 26 file.dvi)
```

It is recommended to use the newer `.sty` file with a PDF backend instead of

doing all this.

Where to find history

You shouldn't need this, but old versions of this utility (along with some non-LaTeX utilities for handling scans and reflowing documents for giant print) may be found on GitHub at https://github.com/ssb22/scan-reflow and on GitLab at https://gitlab.com/ssb22/scan-reflow and on BitBucket https://bitbucket.org/ssb22/scan-reflow and at https://gitlab.developers.cam.ac.uk/ssb22/scan-reflow and in China: https://gitee.com/ssb22/scan-reflow The current version is also

on the author's home page at https://ssb22.user.srcf.net/notes/#latex