Lengths and when to use them

Asked 11 years, 3 months ago Modified 5 years, 11 months ago Viewed 393k times

345

There are many lengths in LaTeX. For instance, `\enskip`, `\enspace`, `\quad`, `\parskip`, `\smallskip`, ... Some of them are mostly used for vertical spacing, some others are mostly used for horizontal spacing, and perhaps some of them are used in both cases. I wonder whether there is a list that indicates the meaning and intended use of each length in LaTeX. By "intended use" I mean not only whether the length is vertical or horizontal, but in which circumstances it is a good practice to use that specific length.

spacing lengths Edit tags

Share Edit Follow Close Flag

edited Jan 19, 2012 at 11:59

asked Jan 18, 2012 at 12:58

ASdeL
4,765 3 19 24

2 Answers

Sorted by: Reset to default Date modified (newest first)

The first thing is to know that there are *spacing parameters* and *spacing commands*; `\parskip` and `\parindent` belong to the former category, `\enskip`, `\quad` and `\smallskip` to the latter.

A complete list of the spacing parameters would be quite long, so let's concentrate on spacing commands.

Vertical spacing commands

- `\smallskip`, `\medskip` and `\bigskip` leave a vertical space of some amount predefined by the class; they are particular cases of `\vspace{<skip>}`; if given in mid paragraph they produce a vertical space between the line where they happen to be set and the following, so they are best used *between* paragraphs.
- `\smallbreak`, `\medbreak` and `\bigbreak` do almost the same, but they remove a preceding vertical space if less than what they would insert; they also terminate a paragraph and tell TeX that they mark a good point where a page break may happen.
- `\addvspace{<skip>}` tries merging with possible other spacing of the same kind, in order to space by the maximum amount of the two (in some occasions this might not happen, the problem is quite subtle).
- `\vfill` is equivalent to `\vspace{\fill}` and tells TeX to fill with white space.
- The variant `\vspace*{<skip>}` tells TeX not to ignore the vertical spacing also if it happens to fall just after a page break; all the previous vertical spacings will in fact disappear at page breaks.

Horizontal spacing commands

- `\enskip`, `\quad`, `\qquad` leave a horizontal space of respectively half an em, one em and two ems. The "em" is a font depending length, frequently as wide as a capital M in the current font.
- `\hspace{<skip>}` is a general horizontal spacing command, that tells TeX to leave that amount of horizontal space.
- `\hspace*{<skip>}` is analogous, but won't disappear at a line break.
- `\hfill` is equivalent to `\hspace{\fill}`.
- `\,` and `\!` leave respectively a thin space and its negative; `\,` can be used to fix some bad spacings caused by visually incompatible pairs of characters (an uppercase letter attached immediately after a lowercase one might be an occasion) or in cases such as `D.\,E.~Knuth` (that somebody prefers to `D.~E.~Knuth`; note that `D.E.~Knuth` is wrong).

What horizontal commands to use? In general it's best to rely on `\quad` and friends, that come from centuries of typography. When a particular application demands a different approach, the `\hspace` command serves the purpose.

I don't recall any spacing command that can be used both for vertical and horizontal spacing.

You mention `\enspace` which is, IIRC, not listed in the LaTeX manual (as aren't `\smallbreak`, `\medbreak` and `\bigbreak`); it's inherited from Plain TeX and is almost the same as `\enskip`, but technically it is a kern, rather than a skip.

The commands `\,` and `\!` have a "long version", which is to be used only in text mode (so not in formulas): `\thinspace` and `\negthinspace`. They, like `\enspace`, insert kerns which are different from skips in that they do not generally define a line break point (the real rule is quite complex, see the TeXbook); in particular TeX won't break a line between "D." and "E." in `D.\,E.~Knuth`. Moreover, kerns have only their natural width (see below for skips).

What's a `<skip>`?

A `<skip>` is a three pronged length specification:

```
<natural width> plus <stretching> minus <shrinking>
```

For example, `\smallskip` is equivalent to `\vspace{3pt plus 1pt minus 1pt}` (in the standard classes) that tells TeX to leave a vertical space of 3pt, but shrinkable up to 2pt or stretchable (optimally) up to 4pt. However, when a stretch component is present, TeX is allowed to stretch that space also beyond the stated specification, in an emergency; this happens frequently when a `\pagebreak` command is found when TeX has not enough material to fill correctly the current page.

The lengths can be specified in any of the legal TeX units of measure, but the stretch and shrink components can be expressed also in terms of *infinite units* `fil`, `fill` and `filll` (to be used with care). For example, if in a page TeX finds `\vfill`, which is the same as `\vspace{0pt plus 1fill}`, all the `\smallskip`s found in the page will be 3pt wide, as the infinite component of `\vfill` *wins*.

The command `\stretch` can be used in this context: `\stretch{<decimal number>}` is equivalent to the skip specification `0pt plus <decimal number>fill`, so `\vspace{\stretch{2}}` is equivalent to say `\vfill\vfill`. TeX will fill with white space proportionally to the `fill` components. So

```
\clearpage
\thispagestyle{empty}
\vspace*{\stretch{1}} % the same as `\vspace*{\fill}`
\begin{flushright}
\itshape
To my dog\\
and my cat
\end{flushright}
\vspace{\stretch{2}}
```

```
\clearpage
```

can be used to have a dedication placed in the page with twice as much white space below than above. Note that above one has to use `\vspace*` in order to avoid the spacing disappearing at the page break.

Caveat

A recent question presented a funny problem: Space generated by theorem labels (XeTeX)

It turns out that this is an undocumented feature (aka "bug that won't be fixed"): the commands `\,` `\!` `\thinspace` `\negthinspace` `\enspace` are defined to be

```
\kern<dimen>
```

so if they are issued in vertical mode (when TeX hasn't yet started a paragraph) they produce a ***vertical*** space rather than a horizontal one.

Be sure to issue them when a paragraph has been started; note, for example, that `\item` *doesn't* start a paragraph, nor does `\begin{<theorem>}` (where `<theorem>` stands for any environment defined with `\newtheorem`).

On the contrary, `\quad`, `\qquad`, `\enskip` and `\hspace` do start a paragraph if TeX is in vertical mode when it finds them.

Share Edit Follow Flag

edited Jun 10, 2020 at 12:32
Community Bot
1

answered Jan 18, 2012 at 14:00
egreg
1.1m 130 2564 4125

I found `\medspace`, `\negmedspace` and friends here: math.dartmouth.edu/~rweber/latex/minitexsymb.pdf. Are those Plain TeX commands also? What about commands like `\vspace{\stretch{2}}` ? I understand that it is equivalent to \vfill\vfill. – ASdeL Jan 18, 2012 at 14:16

2 `\medspace` is undefined in LaTeX; I'll add about `\stretch`. – egreg Jan 18, 2012 at 14:18

57 This sort of answer is why I really love tex.stackexchange.com and it's awesome community. – topskip Jan 18, 2012 at 14:29

1 @Altermundus All the things I analyzed *are* defined in LaTeX. – egreg Jun 7, 2012 at 15:51

4 @tanh This is a point where I strongly disagree with Bringhurst; I can understand thin spaces, but I find "no space" wrong. – egreg Dec 16, 2015 at 13:25

43

Review the following link:

1. Typesetting Math in LaTeX

When we look at spaces in math mode, all spaces are ignored. Thus, `$x y$` is the same thing as `$xy$`. This is very important when typesetting your math formulas, etc. Ways in which you can add spaces include:

- `\,` used as `$x\, y$` which yields a `thinspace`
- `\;` used as `$x\; y$` which yields a `thickspace`
- `\(space)` used as `$x\ y$` which yields a `charspace`
- `\quad` used as `$x\quad y$` which yields a `quadspace`
- `\qquad` used as `$x\qquad y$` which yields a `double quadspace` and
- `\!` used as `$x\! y$` which yields a `negative thinspace`.

Note that in all of the above spaces, a char space is necessary to avoid LateX reading your space command as a `\newcommand` due to the `\`. For example: `$x\, y$` is good and `$x\,y$` is bad.

In math mode, yet, one may use text spaces to generate custom spaces between math inputs but one has to be careful how we input such text. With `amsmath` the command `\text{ }` is recommended. `\mbox{}` can also be used. Other possible implementations of spaces in math mode may include the `\hspace{ }` or `\hphantom{ }` commands.

1. TeXbyTopic Documentation
2. Length Sizes

Note that the TeXbyTopic Documentation is a bit technical and may not be as simple as the first link. I will be providing an example later today.

Share Edit Follow Flag

edited Dec 29, 2012 at 14:56

mhelvens
6,036 2 32 64

answered Jan 18, 2012 at 13:47

azetina
28.3k 20 107 196

8 not true that typed spaces must be input after all these spacing commands. yes, they must be typed after `\quad` (or any command composed of a backslash plus letters), but after a "control character" (a backslash followed by a single non-letter), space isn't necessary, although it will be ignored in math mode. – barbara beeton Jun 7, 2012 at 15:08

I understand but in the past certain compilers ignored such space. I will edit as you suggest madam.:-) – azetina Jun 7, 2012 at 15:11

1 hmmm. i wasn't aware of them being ignored. can you identify the miscreant compilers? (my comment was based on the rules in the texbook.) – barbara beeton Jun 7, 2012 at 15:14

"Note that in all of the above spaces, a char space is necessary to avoid LaTeX reading your space command as a `\newcommand` due to the `\ `. For example: `$x\, y$` is good and `$x\,y$` is bad." I have no idea what that means. Can you explain a bit? – L. F. Dec 29, 2019 at 14:20

2 @L.F. In fact, there is no difference between `$\,y$` and `$x\, y$` as far as I'm concerned. This is because `,` is not a letter and thus TeX doesn't look for a word boundary when trying to find the end of the control sequence name. – FUZxxl Apr 30, 2020 at 15:06