Improving “setspace”

Markus Kohm*

Version 2024-11-12 v1.04

Package setspaceenhanced has started as hack module of the KOMA-Script package scrhack years ago to fix an issue when using package setspace with other document font sizes than 10 pt, 11 pt or 12 pt. This became necessary because package setspace originally only supported these three font sizes and loading the package with a floating point definition of `\@ptsize` even resulted in errors. These two issues has been fixed some years ago. Now, setspace uses a static factor for all font sizes but 10 pt, 11 pt, or 12 pt.

Additionally, if you change font size inside the document setspace still uses the stretch factor of the document font size instead of using a proper stretch factor for the new font size.

Package setspaceenhanced provides improvements for all these limitations and also some additional enhancements.

Contents

*Repository and bug reports: https://github.com/komascript/third-party-enhancements

1 Why should I use this package instead of setpace at least if I use a KOMA-Script class or KOMA-Script package scrextend or a similar package?

From 2006 KOMA-Script classes and KOMA-Script package scrextend provide option `fontsize` for setting document font sizes not limited to 10 pt, 11 pt, or 12 pt. Using this option is also not limited to integer font sizes but also supports floating point sizes like 11.5 pt. To support such font sizes, the specification of macro `\@ptsize` has been changed to be no longer either 0, 1 or 2 but to be the font size minus 10 in pt.

By the way, e.g., Ivan Valbusa's package fontsize adopted this existing definition of `\@ptsize` from KOMA-Script by using most of the font size code of KOMA-Script.

Unfortunately package setspace provides stretch factors for individual font sizes only for 10 pt, 11 pt or 12 pt. For all sizes between these size, it uses the factor of the down rounded integer. For all sizes below 10 pt or 11 pt it uses a static value, 1.25 for `\onehalfspacing` and 1.667 for `\doublespacing`. Package setspaceenhanced uses for `\onehalfspacing` and `\doublespacing` a calculation of the stretch factor depending on the selected baseline skip and font size. With this, every font size is supported.

But that's just the tip of the iceberg and even comparatively unimportant. Much more important are the reasons in the following section, which also apply when using a KOMA-Script class or the KOMA-Script package scrextend or a similar package.

2 Why should I use this package instead of setpace independent from using a KOMA-Script class or KOMA-Script package scrextend or similar packages?

setspace does not care for the `\baselineskip` selected by `\fontsize`. Instead it sets the stretch factor always depending on the document font size. And using `\singlespacing`, `\onehalfspacing` or `\doublespacing` or even `\setstretch` after switching the font size using `\fontsize` instead of `\Huge`, `\huge`, `\LARGE`, `\Large`, `\large`, `\normalsize`, `\small`, `\footnotesize`, `\scriptsize`, `\tiny`, or another command defined using `\@setsize`, reactivates the last used such font size commd. So using something like

```
\normalsize\fontsize{5pt}{7pt}\selectfont\onehalfspacing
```

result in `\normalsize` with onehalfspacing, not 5 pt with onehalfspacing!

Package setspaceenhanced uses for `\onehalfspacing` and `\doublespacing` a calculation of the stretch factor depending on the selected baseline skip and font size. So using something like

```
\normalsize\fontsize{5pt}{7pt}\onehalfspacing
```

will not set the stretch factor based on `\normalsize` but based on font size 5 pt with baseline skip 7 pt. So this results in real onehalfspacing of the 5 pt font.

In other words: Package setspaceenhanced uses a completely different definition of `\onehalfspacing` and `\doublespacing`. It always uses `\f@size` and `\f@baselineskip` to calculate a factor resulting in real onehalfspacing and doublespacing. This also means, if you use one of these commands after changing the font size, a new stretch factor is calculated depending on the current font size without changing the font size.

3 How to use setspaceenhanced?

In the document preamble of your document you just can replace

```
\usepackage{setspace}
```

by

```
\usepackage{setspaceenhanced}
```

to load package setspaceenhanced. This does still also load package setspace but additionally replaces several commands of setspace to avoid the issues shown in section 1 and section 2. setspaceenhanced also does support the same options as setspace. So you can also replace, e.g.

```
\usepackage[onehalfspaceing]{setspace}
```

by

```
\usepackage[onehalfspacing]{setspaceenhanced}
```

If you want you can alternatively also load both packages explicitly, either setspace before setspaceenhanced or—if you want—setspaceenhanced before setspace. In this case, you should use the same options for both package.

This is also useful, if you use a package, that uses setspace itself. In this case, you always should load setspaceenhanced *before* the package, that uses setspace. Otherwise it is very likely that the initial line spacing is still done with the unchanged commands and settings of setspace and therefore the full functionality of setspaceenhanced cannot be reached. Only if setspaceenhanced is loaded before the first use of `\singlespacing`, `\onehalfspacing`, `\doublespacing`, or `\setstretch` can it be ensured that the enhancements of setspaceenhanced are initialized and used correctly.

When using a class that uses setspace, the correct operation can be ensured with

```
\AddToHook{package/setspace/after}{\RequirePackage{setspaceenhanced}}
```

even before `\documentclass`. This requires at least LaTeX 2020/10/01. For older versions of LaTeX you can use

```
\RequirePackage{scrlfile}
\AfterPackage{setspace}{\RequirePackage{setspaceenhanced}}
```

also before `\documentclass`. This would require the KOMA-Script package scrlfile. In both cases you should also use the same optional argument for `\RequirePackage`, that is used for loading setspace.

(added 2024-11-12) Note: If package setspace has been loaded *before* package setspaceenhanced, setspaceenhanced uses the options to package setspace for the initialization. But you can overwrite the initialization by loading setspaceenhancded with options or passing options to setspaceenhanced. On the other hand, package setspace *should never* be loaded with options after loading setspaceenhanced. So

```
\usepackage[doublespacing]{setspace}
\usepackage{setspaceenhanced}
```

would be the same like

```
\usepackage{setspace}
\usepackage[doublespacing]{setspaceenhanced}
```

or like

```
\usepackage[doublespacing]{setspaceenhanced}
\usepackage{setspace}
```

or like

```
\usepackage[doublespacing]{setspaceenhanced}
```

But

```
\usepackage{setspaceenhanced}
\usepackage[doublespacing]{setspace}
```

would fail with an option clash error, because setspaceenhanced already loaded setspace without options.

Package setspaceenhanced provides all options and commands of the user interface of setspace, see [TF22]. Following we document only the differences and enhancements.

There are some options influencing the behavior and result of the examples shown in the section before. All these options are ⟨*key*⟩=⟨*value*⟩ options using the new LaTeX kernel interface. Therefore the package needs at least LaTeX 2022-06-01.

\spacesetup Options can be set as global option via `\documentclass`, as package option via `\usepackage` or using:

`\spacesetup{`⟨*options*⟩`}`

The command can be used in the document preamble and also in the document body. In the document body the changes are local to the current group.

Available options:

`byselectfont` (*opt.*) `byselectfont=`⟨*boolean*⟩ initial=`false`, default=`true`

In the setspaceenhanced examples in the previous section, the correct factor has only been used, because of using `\onehalfspacing` after changing the font size, e.g., to `\small`. If you use `\onehalfspacing` before changing the font size, the factor is calculated with the previous valid font size, which is the document font size 12 pt in all these examples. This behavior can be changed to a more dynamic automatism using option `byselectfont` or `byselectfont=true`. This will use the generic LaTeX hook `selectfont` to reactivate `\onehalfspacing` or `\doublespacing` after every `\selectfont` if the font size has been changed. However note, this is only done if baselineskip is not less than the font size. So for a font selection like `\fontsize{6}{0}\selectfont` the factor recalculation is ignored, because it does not make sense.[1]

\selectfont

! Please note: The capabilities of this option are unfortunately limited. Among other things, it is dependent on the internal function of `\selectfont`. It can therefore lead

[1] In my opinion using a baselineskip less than the font size never makes sense. But LaTeX itself uses baselineskip 0 inside the definition of `\LaTeX`.

to undesirable effects in some cases. In this case, you should disable the option either locally or globally via `\spacesetup{byselectfont=false}` or directly when loading setspaceenhanced.

`onehalfspacing` (*opt.*)

`doublespacing=`⟨*real*⟩ initial=`nan`, default=*empty*

If this option is used without value, it is the same as setspace's package option `doublespacing` or using command `\doublespacing`. But if you assign a real number[2] this would be used as the new stretch factor used for doublespacing. This also means, that the default calculation of the factor is deactivated. But a factor of `nan` would reactivate the calculation of the factor depending on the font size and the baseline skip set for the font size. It is recommended to use the option always without value!

`keepfontsize` (*opt.*)
(changed 2023-09-19)
`\setstretch`

`keepfontsize=`⟨*boolean*⟩ initial=`false`, default=`true`

As explained in section 2, setspace's `\setstretch` behaves different after a font size command like `\Huge`, `\huge`, `\LARGE`, `\Large`, `\large`, `\normalsize`, `\small`, `\footnotesize`, `\scriptsize`, `\tiny`, or another command defined using `\@setsize` than after using `\fontsize{...}{...}\selectfont`. With the last the font size will be reset to the previous usage of one of the other or the document font size. For a lot of users this is somehow unexpected. With option `keepfontsize` or `keepfontsize=true` this is changed and using `\setstretch` does not reactivate the last used `\Huge`, `\huge`, `\LARGE`, `\Large`, `\large`, `\normalsize`, `\small`, `\footnotesize`, `\scriptsize`, `\tiny`.

`onehalfspacing` (*opt.*)

`onehalfspacing=`⟨*real*⟩ initial=`nan`, default=*empty*

If this option is used without value, it is the same as setspace's package option `onehalfspacing` or using command `\onehalfspacing`. But if you assign a real number[2] this would be used as the new stretch factor used for onehalfspacing. This also means, that the default calculation of the factor is deactivated. But a factor of `nan` would reactivate the calculation of the factor depending on the font size and the baseline skip set for the font size. It is recommended to use the option always without value!

`singlespacing` (*opt.*)

`singlespacing=`⟨*real*⟩ initial=1, default=*empty*

If this option is used without value, it is the same as setspace's package option `singlespacing` or using command `\singlespacing`. But if you assign a real number[2] this would be used as the new stretch factor used for singlespacing. So this is similar to using `\SetSinglespace{`⟨*real*⟩`}\singlespacing`. A factor of `nan` would activate the calculation of the factor depending on the font size and the baseline skip set for the font size. It is recommended to use the option always without value!

`nodisplayskipstretch` (*opt.*)
(added 2024-11-12)

`nodispayskipstretch` Same as with setspace. The option turns off the stretching of the space before and after displays using `\setdisplayskipstretch{1.0}`.

Compatibility Notes:

scrhack If you want to use this package together with package scrhack from KOMA-Script before version 3.42, you should deactivate the `setspace` hack using scrhack' option `setspace=false`. From version 3.42 scrhack does not use the old hacks any longer but setspaceenhanced and is therefore compatible again.

[2] Here are all values allowed, that would be allowed as second argument of LaTeX3 function `\fp_set:Nn`

4 How does the result of setspaceenhanced differ from setspace even for using the standard font sizes of the standard classes?

For example if you have a document:

```
\documentclass[12pt]{article}
\usepackage{setspace}
\begin{document}
\small\onehalfspacing This is font size
\csname f@size\endcsname pt with
normal baseline skip \csname f@baselineskip\endcsname.
The current stretch factor is \baselinestretch. This
results in a baseline skip of \the\baselineskip.
\end{document}
```

this will result in:

> This is font size 10.95pt with normal baseline skip 13.6pt. The current stretch factor is 1.241. This results in a baseline skip of 16.87756pt.

But one moment: $10.95\,\text{pt} \cdot 1.5 = 16.425\,\text{pt}$. So the factor seems to be wrong. It is not real onehalfspacing depending on the used font size. It is also not onehalfspacing depending on the document font size, because this would need a baseline skip of 18 pt. So what is it? It is using the stretch factor of 12 pt for the 10.95 pt of `\small`.

With setspaceenhanced:

```
\documentclass[12pt]{article}
\usepackage{setspaceenhanced}
\begin{document}
\small\onehalfspacing This is font size
\csname f@size\endcsname pt with
normal baseline skip \csname f@baselineskip\endcsname.
The current stretch factor is \baselinestretch. This
results in a baseline skip of \the\baselineskip.
\end{document}
```

the result changes:

> This is font size 10.95pt with normal baseline skip 13.6pt. The current stretch factor is 1.207720046225135. This results in a baseline skip of 16.42496pt.

Here the difference from the correct value 16.425 pt is very, very small: 0.00004 pt. So you can say, this is really onehalfspacing depending on the used font size.

Moreover if you have a document:

```
\documentclass[12pt]{article}
\usepackage{setspace}
\begin{document}
\fontsize{5pt}{7pt}\selectfont\onehalfspacing This is font size
\csname f@size\endcsname pt with
normal baseline skip \csname f@baselineskip\endcsname.
```

```
The current stretch factor is \baselinestretch. This
results in a baseline skip of \the\baselineskip.
\end{document}
```

this result in:

> This is font size 12pt with normal baseline skip 14.5pt. The current stretch factor is 1.241. This results in a baseline skip of 17.99446pt.

But

```
\documentclass[12pt]{article}
\usepackage[keepfontsize]{setspaceenhanced}
\begin{document}
\fontsize{5pt}{7pt}\selectfont\onehalfspacing This is font size
\csname f@size\endcsname pt with
normal baseline skip \csname f@baselineskip\endcsname.
The current stretch factor is \baselinestretch. This
results in a baseline skip of \the\baselineskip.
\end{document}
```

results in:

> This is font size 5pt with normal baseline skip 7.0pt. The current stretch factor is 1.071428571428571. This results in a baseline skip of 7.49998pt.

In my opinion this is more the expected result. See the previous section 3 for more information about options like `keepfontsize`.

5 Implementation

We use the new LaTeX kernel feature of ⟨*key*⟩=⟨*value*⟩ options introduced in [TLT22]. So we need at least LaTeX 2022-06-01:

```
\ifnum 0=\ifcsname IfFormatAtLeastTF\endcsname
    \IfFormatAtLeastTF{2022-06-01}{1}{0}
  \else
    0
  \fi\relax
  \PackageError{setspaceenhanced}{LaTeX~kernel~too~old}{
    The~package~needs~at~least~LaTeX~2022-06-01.\MessageBreak
    This~error~is~fatal.~Loading~will~be~aborted
  }
  \endinput
\fi
```

We do not pass any options to setspace, because we handle them different. So we can just load the package here:

```
\RequirePackage{setspace}
```

`\c_@@_single_fp` (*const.*)
`\c_@@_onehalf_fp` (*const.*)
`\c_@@_double_fp` (*const.*)

The internal constants store the absolute factor for singlespacing, onehalfspacing and doublespacing used to calculate the stretch factors.

```
\fp_const:Nn \c_@@_single_fp  { 1.2 }
\fp_const:Nn \c_@@_onehalf_fp { 1.5 }
\fp_const:Nn \c_@@_double_fp  { 2.0 }
```

`\g_@@_single_factor_fp` (*var.*)
`\g_@@_onehalf_factor_fp` (*var.*)
`\g_@@_double_factor_fp` (*var.*)

The internal variables used to store the configured stretch factors for singlespacing, onehalfspacing and doublespacing. If `nan` `\@@_linespread` is calculated.

```
\fp_new:N \g_@@_single_factor_fp
\fp_set:Nn \g_@@_single_factor_fp { 1.0 }
\fp_new:N \g_@@_onehalf_factor_fp
\fp_set_eq:NN \g_@@_onehalf_factor_fp \c_nan_fp
\fp_new:N \g_@@_double_factor_fp
\fp_set_eq:NN \g_@@_double_factor_fp \c_nan_fp
```

`\g_@@_linespread_fp` (*var.*)
`\g_@@_fp` (*var.*)

Storage of the current calculated stretch factor and the used constant.

```
\fp_new:N \g_@@_linespread_fp
\fp_set:Nn \g_@@_linespread_fp { 1.0 }
\fp_new:N \g_@@_fp
```

`\@@_set_spacing:nn` (*intern*)

This function is used to set the stretch factor for one of the spacings.

```
\cs_new:Nn \@@_set_spacing:nn
  {
    \tl_if_blank:nF { #1 }
      {
        \fp_set:cn { g_@@_#2_factor_fp } { #1 }
      }
    \fp_if_nan:nTF { \tl_use:c { g_@@_#2_factor_fp } }
      {
        \fp_set_eq:Nc \g_@@_fp { c_@@_#2_fp }
        \@@_calc_stretch:
      }
      {
        \fp_set_eq:Nc \g_@@_linespread_fp { g_@@_#2_factor_fp }
        \fp_set_eq:NN \g_@@_fp \c_nan_fp
      }
      \setstretch{ \fp_to_decimal:N \g_@@_linespread_fp }
  }
```

`\@@_calc_stretch:` (*intern*)

This macro is used to (re-)calculate the stretch factor `\@@_calcstretch` if the currently used constant is not `nan`.

```
\cs_new:Nn \@@_calc_stretch:
  {
    \fp_if_nan:nF { \g_@@_fp }
      {
        \fp_compare:nNnTF { \dim_to_fp:n \f@baselineskip } < \f@size
          {
            \fp_set_eq:NN \g_@@_fp \g_@@_linespread_fp
          }
          {
            \fp_set:Nn \g_@@_linespread_fp
              {
                \f@size / \dim_to_fp:n { \f@baselineskip } * \g_@@_fp
              }
          }
      }
  }
```

`\selectfont` We also need to hook into `\selectfont` using the general `selectfont` hook to re-calculate the stretch factor after the font size has been changed and `byselectfont=true`.

```
\hook_gput_code:nnn { selectfont } { setspaceenhanced }
  {%
    \bool_if:NT \g_@@_byselectfont_bool
      {
        \cs_if_exist:NT \size@update
          {
            \fp_compare:nNnT \g_@@_linespread_fp = \f@linespread
              {
                \@@_calc_stretch:
                \set@fontsize
                  {
                    \fp_to_decimal:N \g_@@_linespread_fp
                  }
                  \f@size \f@baselineskip
              }
          }
      }
  }
```

`\setstretch` If `keepfontsize=false`, we use the original functionality of setspace. But with `keepfontsize=true` we use `\linespread`:

```
\renewcommand*{\setstretch}[1]{
  \bool_if:NTF \g_@@_fontsize_bool
    {
      \linespread{#1}\selectfont
    }
    {
      \def\baselinestretch{#1}%
      \@currsize
    }
}
```

Almost the end of the package we define all options:

```
\DeclareKeys{%
  byselectfont   .bool_set:N  = \g_@@_byselectfont_bool,
  byselectfont   .default:n   = true,
  byselectfont   .initial:n   = false,
  byselectfont   .usage       = general,
  keepfontsize   .bool_set:N  = \g_@@_fontsize_bool,
  keepfontsize   .default:n   = true,
  keepfontsize   .initial:n   = false,
  keepfontsize   .usage       = general,
  singlespacing  .code        = \@@_set_spacing:nn {#1} {single},
  singlespacing  .usage       = general,
  singlespacing  .initial:n   = ,
  onehalfspacing .code        = \@@_set_spacing:nn {#1} {onehalf},
  onehalfspacing .usage       = general,
  doublespacing  .code        = \@@_set_spacing:nn {#1} {double},
  doublespacing  .usage       = general,
```

```
  nodisplayskipstretch .code  = \setdisplayskipstretch{1.0},
  nodisplayskipstretch .usage = general,
}
```

Now, test for the options used by setspace and use them also for the initialization of the keys above. We do it option by option instead of processing all setspace options, because we want to process only those options setspaceenhanced also knows.

```
\clist_map_inline:nn
  { nodisplayskipstretch, singlespacing, onehalfspacing, doublespacing }
  {
    \@ifpackagewith {setspace} {#1} {\SetKeys{#1}} { }
  }
```

Last but not least process the package options:

```
\ProcessKeyOptions\relax
```

`\singlespacing` `\onehalfspacing` `\doublespacing` `\SetSingleSpace`

We have to redefine the user interface commands for singlespacing, onehalfspacing and doublespacing.

```
\renewcommand*{\singlespacing}{
  \@@_set_spacing:nn {} {single}
}
\renewcommand*{\onehalfspacing}{
  \@@_set_spacing:nn {} {onehalf}
}
\renewcommand*{\doublespacing}{
  \@@_set_spacing:nn {} {double}
}
\renewcommand*{\SetSinglespace}[1]{
  \fp_set:Nn \g_@@_single_factor_fp { #1 }
}
```

`\setspace@singlespace` (*intern*)

Note: Defining this internal macro does not work using setspaceenhanced. Should we add a test for users and package authors, who do not use `\SetSinglespace` but redefine the internal macro?

`spacesetup`

User interface to not need to use `\SetKeys`:

```
\newcommand*{\spacesetup}{\SetKeys[setspaceenhanced]}
```

References

[Car+22] David Carlisle et al. *setspace — Set space between lines*. Version 6.7b. Provides support for setting the spacing between lines in a document. Package options include `singlespacing`, `onehalfspacing`, and `doublespacing`. Alternatively the spacing can be changed as required with the `\singlespacing`, `\onehalfspacing`, and `\doublespacing` commands. Other size spacings also available. Dec. 4, 2022. URL: https://ctan.org/pkg/setspace (visited on 07/25/2023).

[Koh23a] Markus Kohm. *KOMA-Script — A bundle of versatile classes and packages*. Version 3.41. The KOMA-Script bundle provides replacements for the article, report, and book classes with emphasis on typography and versatility. There is also a letter class. July 7, 2023. URL: https://ctan.org/pkg/koma-script (visited on 07/14/2023).

[Koh23b] Markus Kohm. *scrlfile – Installation control (not only) for KOMA-Script packages*. Version 3.41. The package provides hooks for the execution of commands before or after loading files, classes or packages independent from the LaTeX kernel version. July 7, 2023. URL: https://ctan.org/pkg/scrlfile (visited on 07/19/2023).

[TF22] Geoffrey Tobin and Robin Fairbairns. *The setspace Package*. Dec. 4, 2022. URL: https://ctan.org/tex-archive/macros/latex/contrib/setspace (visited on 07/25/2023).

[TLT22] The LaTeX Project Team. "Issue 35." In: *LaTeX News* (June 2022). URL: http://mirrors.ctan.org/macros/latex/base/ltnews35.pdf (visited on 07/14/2023).

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

Change History