Improving “setspace”

Markus Kohm*

Version 2023-10-09 v1.02

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

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? 2

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? 2

3 How to use setspaceenhanced? 3

4 How does the result of setspaceenhanced differ from setpace even for using the standard font sizes of the standard classes? 5

5 Implementation 6

References 9

Index 10

Change History 10

*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 *setpace* 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 \.oneshalfspacing
and 1.667 for \.doublespacing. Package *setspaceenhanced* uses for \.oneshalfspacing 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?

*setpace* 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,
\oneshalfspacing 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, reac-
tivates the last used such font size cmd. So using something like

\normalsize\fontsize{5pt}{7pt}\selectfont\oneshalfspacing

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

Package *setspaceenhanced* uses for \oneshalfspacing 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}\oneshalfspacing

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
\oneshalfspacing and \doublespacing. It always uses \fsize 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.

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 \langle key\rangle=(\langle value\rangle) options using the new \LaTeX kernel interface. Therefore the package needs at least \LaTeX 2022-06-01.

\spacesetup \langle options\rangle

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

\spacesetup{(\langle options\rangle)}

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 \setspace\enhanced\ examples in the previous section, the correct factor has only been used, because of using \texttt{\oneshaispacing} after changing the font size, e.g., to \texttt{\small}. If you use \texttt{\oneshaispacing} 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 automation using option \texttt{byselectfont} or \texttt{byselectfont=true}. This will use the generic \LaTeX{} hook \texttt{selectfont} to reinitialize \texttt{\oneshaispacing} or \texttt{\doublespacing} after every \texttt{\selectfont} if the font size has been changed.

**onehalfspacing (opt.)**

\oneshaispacing\={\real} \ initial=nan, \ default=empty

If this option is used without value, it is the same as \texttt{\setspace}'s package option \texttt{doublespacing} or using command \texttt{\doublespacing}. But if you assign a real number\(^1\) 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 \texttt{nan} would reinitialize 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.)**

\keepfontsize\={boolean} \ initial=false, \ default=true

As explained in section 2, \texttt{\setspace}'s \texttt{\setstretch} behaves different after a font size command like \Huge, \huge, \LARGE, \Large, \large, \normalsize, \small, \footnotesize, \scriptsize, \tiny, or another command defined using \texttt{\@setsize} than after using \texttt{\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 \texttt{keepfontsize} or \texttt{keepfontsize=true} this is changed and using \texttt{\setstretch} does not reactivate the last used \Huge, \huge, \LARGE, \Large, \large, \normalsize, \small, \footnotesize, \scriptsize, \tiny.

**onehalfspacing (opt.)**

\oneshaispacing\={\real} \ initial=nan, \ default=empty

If this option is used without value, it is the same as \texttt{\setspace}'s package option \texttt{onehalfspacing} or using command \texttt{\oneshaispacing}. 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 \texttt{nan} would reinitialize 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 \texttt{\setspace}'s package option \texttt{singlespacing} or using command \texttt{\singlespacing}. But if you assign a real number\(^1\) this would be used as the new stretch factor used for singlespacing. So this is similar to using \texttt{\SetSinglespace{\langle\real\rangle}\singlespacing}. A factor of \texttt{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!

\footnote{Here are all values allowed, that would be allowed as second argument of \LaTeX{} function \texttt{\fp_set:Nn}}
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.

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:

```latex
\documentclass[12pt]{article}
\usepackage{setspace}
\begin{document}
\small\onemilespacing 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 pt · 1.5 = 16.425 pt. So the factor seems to be wrong. It is not real `onemilespacing` depending on the used font size. It is also not `onemilespacing` 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**:

```latex
\documentclass[12pt]{article}
\usepackage{setspaceenhanced}
\begin{document}
\small\onemilespacing 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 `onemilespacing` 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 $\langle$key$\rangle$=$\langle$value$\rangle$ options introduced in [TLT22]. So we need at least \LaTeX\ 2022-06-01:

\begin{verbatim}
1 \ifnum 0=\ifcsname IfFormatAtLeastTF\endcsname
2 \IfFormatAtLeastTF{2022-06-01}{1}{0}
3 \else
4 0
5 \fi\relax
6 \PackageError{setspaceenhanced}{LaTeX"kernel"too"old}{The \textquote{package} needs \textquote{at least \LaTeX\ 2022-06-01.} \MessageBreak This \textquote{error} is \textquote{fatal}. \MessageBreak \textquote{Loading} will \textquote{be} aborted}
7 \endinput
8 \fi
9 \fi
10 \fi
11 \fi
\end{verbatim}
We do not pass any options to \texttt{setspace}, because we handle them different. So we can just load the package here:

\begin{verbatim}
\usepackage{setspace}
\end{verbatim}

\begin{verbatim}
\c@@single_fp (const.)  The internal constants store the absolute factor for singlespacing, onehalfspacing and doublespacing used to calculate the stretch factors.
\c@@onehalf_fp (const.)  \begin{verbatim}
\fp_const:Nn \c@@single_fp { 1.2 }
\end{verbatim}
\c@@double_fp (const.)  \begin{verbatim}
\fp_const:Nn \c@@onehalf_fp { 1.5 }
\end{verbatim}
\c@@double_fp (const.)  \begin{verbatim}
\fp_const:Nn \c@@double_fp { 2.0 }
\end{verbatim}
\end{verbatim}

\begin{verbatim}
\g@@single_factor_fp (var.)  The internal variables used to store the configured stretch factors for singlespacing, onehalfspacing and doublespacing. If \texttt{nan} \@linespread is calculated.
\g@@onehalf_factor_fp (var.)  \begin{verbatim}
\fp_new:N \g@@single_factor_fp
\fp_set:Nn \g@@single_factor_fp { 1.0 }
\end{verbatim}
\g@@double_factor_fp (var.)  \begin{verbatim}
\fp_new:N \g@@onehalf_factor_fp
\fp_set_eq:NN \g@@onehalf_factor_fp \c_nan_fp
\end{verbatim}
\g@@double_factor_fp (var.)  \begin{verbatim}
\fp_new:N \g@@double_factor_fp
\fp_set_eq:NN \g@@double_factor_fp \c_nan_fp
\end{verbatim}
\end{verbatim}

\begin{verbatim}
\g@@linespread_fp (var.)  Storage of the current calculated stretch factor and the used constant.
\g@@fp (var.)  \begin{verbatim}
\fp_new:N \g@@linespread_fp
\fp_new:N \g@@fp
\end{verbatim}
\end{verbatim}

\begin{verbatim}
\@@set_spacing:nn This function is used to set the stretch factor for one of the spacings.
\begin{verbatim}
\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 }
}
\end{verbatim}
\end{verbatim}

\begin{verbatim}
\@@calc_stretch: This macro is used to (re-)calculate the stretch factor \texttt{\@@calcstretch} if the currently used constant is not \texttt{nan}.
\begin{verbatim}
\cs_new:Nn \@@calc_stretch:
{ \fp_if_nan:nF { \g@@fp } 
{ \fp_set:Nn \g@@linespread_fp }
}
\end{verbatim}
\end{verbatim}
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.

\selectfont \hook_gput_code:nnn { selectfont } { setspaceenhanced }
{
\bool_if:NT \g_@@_byselectfont_bool
{
\cs_if_exist:NT \size@update
{
\fp_set:Nn \l_tmpa_fp { \f@linespread }
\fp_compare:nNnT \g_@@_linespread_fp = \l_tmpa_fp
{
\@@_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{
singlespacing .code = \@@_set_spacing:nn {#1} {single},
singlespacing .usage = general,
onehalfspacing .code = \@@_set_spacing:nn {#1} {onehalf},
onehalfspacing .usage = general,
doublespacing .code = \@@_set_spacing:nn {#1} {double},
doublespacing .usage = general,
byselectfont .bool_set:N = \g_@@_byselectfont_bool,
And process them:
\ProcessKeyOptions\relax
\singlespacing
\onehalfspacing
\doublespacing
\SetSingleSpace

\setspace@singlespace
\setspace@onehalfspacing
\setspace@doublespacing

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. \textit{setspace} — \textit{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: \url{https://ctan.org/pkg/setspace} (visited on 07/25/2023).


### 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.

#### B
- `byselectfont (opt.)` ........... 4

#### C
- `\c_@@_double_fp (const.)` 13
- `\c_@@_onehalf_fp (const.)` 13
- `\c_@@_single_fp (const.)` 13

#### Commands:
- `\doublespacing` .... 96
- `\onehalfspacing` .... 96
- `\selectfont` .... 4, 52
- `\SetSingleSpace` .... 96
- `\setstretch` .... 2, 4, 71
- `\oneshowspace` .... 96
- `\spacesetup` .... 3, 108

#### Constants:
- `\c_@@_double_fp` .... 13
- `\c_@@_onehalf_fp` .... 13
- `\c_@@_single_fp` .... 13

#### D
- `\doublespacing` .... 96
- `\selectfont` .... 4, 52

#### G
- `\g_@@_double_factor_fp`
  `(var.)` .... 16
- `\g_@@_fp`
  `(var.)` .... 22
- `\g_@@_linespread_fp`
  `(var.)` .... 22
- `\g_@@_onehalf_factor_fp`
  `(var.)` .... 16
- `\g_@@_single_factor_fp`
  `(var.)` .... 16
- `\SetSingleSpace` .... 96
- `\setstretch` .... 2, 4, 71
- `\singlesspace` .... 96
- `\singlespace (opt.)` .... 4
- `\spacesetup` .... 3, 108

#### K
- `keepfontsize (opt.)` .... 4

#### O
- `\onehalfspacing` .... 96
- `\oneshowspace (opt.)` .... 4

#### T
\TeX macros (internal):
- `\@@_calc_stretch:` .... 42
- `\@@_set_spacing` .... 45
- `\setstretch\nspace=setup` .... 108

#### V
- Variable:
- `\g_@@_double_factor_fp` .... 16
- `\g_@@_fp` .... 22
- `\g_@@_linespread_fp` .... 22
- `\g_@@_single_factor_fp` .... 16

### Change History

**v0.1 – 2023/06/01**

General: new KOMA-Script spin-off .... 1

**v1.0 – 2023/08/04**

General: release .......................... 1
General: option fontsize renamed to keepfontsize because of incompatibility (issue #1) 
\g_{00\_double\_factor\_fp}: missing prefix g added to variable name  
\g_{00\_fp}: missing prefix g added to variable name