

The `scalebar` package

Michael Lake
mikel@speleonics.com.au

v1.0 – 2003/05/01

1 Introduction

This document describes the `scalebar` package for $\text{\LaTeX} 2_{\epsilon}$, which creates scalebars for maps, diagrams or photos. It was designed for use with cave maps, but can be used for anything from showing a scalebar in kilometers for topographic maps to a scalebar in micrometers for an electron microscope image.

Here is an example scalebar. See the `scalebar_examples` file for further examples and possible uses.



2 Usage

`\scalebar` The `scalebar` package defines one user command. To use the command you write;

```
\scalebar[<inverse>]{<length>}{<minordivs>}{<majordivs>}  
          {<starting No.>}{<ending No.>}{<units>}
```

where the six mandatory arguments are:

<i><length></i>	the desired length of the scalebar on paper e.g. 10cm or 4in
<i><minordivs></i>	number of minor divisions within the first major division e.g. 4 (the first major division will always be subdivided unless this value is set to 1)
<i><majordivs></i>	number of major divisions e.g. 5
<i><starting No.></i>	the number that the scalebar text will start from e.g. 0 or -0.5
<i><ending No.></i>	the number that the scalebar text will end with e.g. 2.5 or 25
<i><units></i>	the units for the scalebar text e.g. μm or km.

The optional argument *<inverse>* reverses the black and white regions. The default setting is for the top of the first bar to be black.

3 Current Limitations and Future Enhancements

Scalebar height/text ratio is hardcoded The height of the black rules in the scalebars is `1.2ex` so it scales with the current font height. This value seems about right but if you want a different height compared to the text height you will have to modify the style file (modify `\SB@Height`). This could possibly be set as an optional argument to the package.

Thickness of the thin rule is fixed The thickness of the thin rule is set to 0.2mm. This seems about right. If you wish to change it is easy to modify the style file (`\SB@Thick`).

Number of decimal places is limited to one Scalebars rarely display more than one decimal place in the scalebar text. This package rounds the decimals of the displayed text to one place. If you want two decimal places you will have to modify the style file.

Please let me know if you encounter any problems. If you have suggestions for extra options to add, then code for that would be appreciated.


4 Implementation

First we have to decide how a scalebar is constructed. Looking at a scalebar one can see that it is made up of alternating black and white bars. It would seem sensible therefore to define some sort of “unit” or “building block” that can be iterated any number of times to create a scalebar of arbitrary length and with an arbitrary number of divisions. The diagram below of a scalebar and its deconstruction shows one way to break it up into units.



Looking at the parts above you can see there is a thin vertical rule at the start of each scalebar. This is followed by repeating “scalebar units” each one of which alternates in color. Finally there is a thin vertical rule to close the end of the scalebar.

I have therefore defined two scalebar units. The first one typesets the thick black rule on the top. After writing this rule we then move back horizontally a distance equal to the width of the rule written. Then we write the thinner rule underneath it by using a `raisebox` with a negative vertical distance. This unit looks like this:

 X (the X shows the position of the baseline)

The second type of unit has the thick black rule on the bottom like this:

 X

By stringing these two types of units together within a while loop we can create our scalebars. Let’s now proceed to the start of the macro.

4.1 Required Packages

We have used the `fp` package to provide floating point calculations and a rounding function via the `\FPupn` macro. The `calc` package provides an easy calculation syntax for simple subtraction of lengths. The `ifthen` package provides the `whiledo` loop and the `ifthenelse` construct.

4.2 Initialisation

Note that we have to place a `%` at the end of each command to suppress any whitespace. Otherwise the scalebar will “break apart”.

`counters/lengths` Define counters and lengths.

The first three mandatory arguments specify the length of the scalebar and the numbers of the divisions. The arguments that specify the format of text underneath the scalebar will be introduced later.

```
1 \newlength{\SB@Length}%
2 \newcounter{SB@majordivs}%
3 \newcounter{SB@minordivs}%
```

These are some further general counters and lengths.

```
4 \newcounter{SB@evenodd}%
5 \newcounter{SB@countup}%
6 \newlength{\SB@Height}%
7 \newlength{\SB@Thick}%
8 \newlength{\SB@MajorWidth}%
9 \newlength{\SB@MinorWidth}%
10 \newlength{\SB@DivisionWidth}%
11 \newlength{\SB@TextWidth}%
```

Now define two small commands to typeset the two ‘scalebar units’.

`\SB@unitT` This command typesets the unit with the thick black rule on the *top*.

```
12 \newcommand{\SB@unitT}{%
13 \rule{\SB@DivisionWidth}{\SB@Height}\hspace{-\SB@DivisionWidth}%
14 \raisebox{-\SB@Height}{\rule{\SB@DivisionWidth}{\SB@Thick}}}%
```

`\SB@unitB` This command typesets the thick black rule on the *bottom*.

```
15 \newcommand{\SB@unitB}{%
16 \raisebox{\SB@Height-\SB@Thick}%
17 {\rule{\SB@DivisionWidth}{\SB@Thick}}\hspace{-\SB@DivisionWidth}%
18 \raisebox{-\SB@Height}{\rule{\SB@DivisionWidth}{\SB@Height}}}%
```

4.3 The scalebar macro

`\scalebar` Start the scalebar command and process the command arguments.

args The first argument is an optional argument which is set to nothing as default. Normally the value of `SB@evenodd` is 0 which will result in the black rule of the first division being on top. If `\inverse` is set the black rule will be set on the bottom.

```
19 \newcommand{\scalebar}[7] [] {%
20 %
21 \ifthenelse{\equal{\inverse}{#1}}{%
22 {\setcounter{SB@evenodd}{1}}%
23 {\setcounter{SB@evenodd}{0}}%
```

Read in the next three mandatory arguments.

```
24 \setlength{SB@Length}{#2}%
25 \setcounter{SB@minordivs}{#3}%
26 \setcounter{SB@majordivs}{#4}%
```

Define the number at which the text will start by `SB@StartNo`; the number at which it will end by `SB@EndNo` and the units the text represents by `SB@TextUnits`. These are set to the last three mandatory arguments.

```
27 \def\SB@StartNo{#5}%
28 \def\SB@EndNo{#6}%
29 \def\SB@TextUnits{#7}%
```

Set the height of the scalebar and thickness of the thin enclosing rule.

Let the height of the thick black rule of a division be `SB@Height` and set it to a value related to the x-height of the current font. This is because we want the scalebar to increase in height if the user specifies a change in font size. It's set after the command begins rather than earlier as we want it to pickup a font change immediately before the scalebar command if any. The thin rule that encloses the divisions (`SB@Thick`) will be hard coded to 0.2mm. This was chosen as it looked about right :-)

```
30 \setlength{SB@Height}{1.2ex}%
31 \setlength{SB@Thick}{0.2mm}%
```

Calculate the lengths of the divisions.

Let's define `SB@MajorWidth` to be the width (i.e. length) of a major division. To calculate this we take the total scalebar length and divide by the number of major divisions. To calculate the length of a minor division, `SB@MinorWidth`, we divide the major division length just calculated by the number of minor divisions.

```
32 \setlength{SB@MajorWidth}{\SB@Length / \theSB@majordivs}%
33 \setlength{SB@MinorWidth}{\SB@MajorWidth / \theSB@minordivs}%
```

4.4 Drawing the scalebar

Now we can start placing ink to paper. Place onto the page the thin vertical rule at the start of the scalebar. As it starts at the baseline we have to make it twice the height of `\SB@Height` and lower it by `\SB@Height`.

```
34 \raisebox{-\SB@Height}{\rule{\SB@Thick}{2\SB@Height}}%
```

Use a while loop to place the minor divisions onto the page. (Use a new counter rather than decrementing `minordivs` variable.) Note that we alternate between using the `\SB@unitT` or the `\SB@unitB` unit depending on the value of the `SB@evenodd` variable. That value is set at the beginning and depends on the setting of the optional *inverse* argument.

```
35 \setlength{\SB@DivisionWidth}{\SB@MinorWidth}%
36 \setcounter{SB@countup}{0}%
37 \whiledo{\not\theSB@countup=\theSB@minordivs}{%
38 \ifthenelse{\isodd{\value{SB@evenodd}}}{\SB@unitB}{\SB@unitT}%
39 \addtocounter{SB@evenodd}{1}%
40 \addtocounter{SB@countup}{1}}%
```

Now that the minor divisions are done we can write the major divisions.

Note that we have to reset the length of the rule from the current width of a minor division to the width of a major division. Set the counter this time to start at 1 rather than 0 as we have already written all the minor divisions which adds up to one major division.

```
41 \setlength{\SB@DivisionWidth}{\SB@MajorWidth}%
42 \setcounter{SB@countup}{1}%
43 \whiledo{\not\theSB@countup=\theSB@majordivs}{%
44 \ifthenelse{\isodd{\value{SB@evenodd}}}{\SB@unitB}{\SB@unitT}%
45 \addtocounter{SB@evenodd}{1}%
46 \addtocounter{SB@countup}{1}}%
```

Finally we have to print the thin vertical rule at the end.

```
47 \raisebox{-\SB@Height}{\rule{\SB@Thick}{2\SB@Height}}%
```

4.5 Typesetting the scalebar numbers

Typesetting of the text underneath the scalebar requires three pieces of information; the number at which the text will start, the number at which it will end and what units the text represents. The numbering does not have to start at zero.

Some scalebars will consist of integers only as in the text below.

0 2 4 6 8 10 km

Other scalebars will have decimals. Although usually scalebars only show one decimal place at most.

-0.5 0 1.5 2.0 2.5 cm

We now use macros provided by the `fp.sty` (fixed point) package to perform some simple arithmetic and rounding.

Calculate how much we will increment the numbers by subtracting the starting and ending numbers (`SB@StartNo` and `SB@EndNo`) then dividing their difference by the number of major divisions.

```
48 \FPupn\SBIncrement%
49 {\the\value{SB@majordivs} \SB@StartNo{} \SB@EndNo{} - /}%
```

Now we need to work out how many decimal places to display for the numbers.

Start off with rounding set to none. If the calculated increment is an integer then don't do anything, otherwise set rounding on and round the increment to one decimal place. This would suffice for many variations of starting and ending number, however if the user entered -0.5 to 2.5 for these values respectively then the increment (3.0) would be an integer and the rounding would be set to zero. In these cases though the starting and ending numbers must be non-integer thus by testing all three numbers we can cover all cases.

```
50 \def\SBRound{0}%
51 \FPifint\SBIncrement%
52 \else\def\SBRound{1}\FPupn\SBIncrement{\SBIncrement{} 1 round}\fi%
53 \FPifint\SB@StartNo%
54 \else\def\SBRound{1}\FPupn\SB@StartNo{\SB@StartNo{} 1 round}\fi%
55 \FPifint\SB@EndNo%
56 \else\def\SBRound{1}\FPupn\SB@EndNo{\SB@EndNo{} 1 round}\fi%
```

Now remember that we are at the end of the scalebar so to print the text underneath we have to move back to the left and down. Move left by a distance equal to the scalebar length (`SB@Length`) plus a little bit more to take into account the thickness of the thin line at the start and end of the scalebar (`SB@Thick`). Then move down by enough to place the text at a nice distance below the scalebar. Remember that the distance `SB@Height` depends on the current font in a fixed ratio set earlier.

```
57 \hspace{-\SB@Length}\hspace{-\SB@Thick}%
58 \raisebox{-3\SB@Height}{%
```

Now we are ready to print the numbers. The length `SB@TextWidth` contains the width of the text that we are about to put to page. We need to calculate this so that we can fine position the center of the text to be exactly under the start of each major division. Note how we use the value of `SBRound` to set the number of decimal places in the text we typeset.

```
59 \FPset\SBNextNo\SB@StartNo%
60 \setcounter{SB@countup}{0}%
61 \whiledo{\not\theSB@countup>\theSB@majordivs}{%
62 \FPupn\SBNextNo{\SBNextNo{} \SBRound{} round}%
63 \settowidth{\SB@TextWidth}{\SBNextNo}%
64 \hspace{-0.5\SB@TextWidth}%
65 \SBNextNo\hspace{-0.5\SB@TextWidth}\hspace{\SB@MajorWidth}%
66 \FPupn\SBNextNo{\SBNextNo{} \SBIncrement{} add}%
67 \addtocounter{SB@countup}{1}}%
```

Now all the numbers are on the page we are nearly ready to append the scalebars units to the end. First though we have to move back by the distance

`\SB@MajorWidth` as the last while loop added this unwanted space, then forward half of the width of the last number printed. Finally add a thin space and the units and finish the `scalebar` command with its closing brace.

```
68 \hspace{-\SB@MajorWidth}\hspace{0.5\SB@TextWidth}\,\SB@TextUnits}%
69 }%
```

That's the end of the macro.

License

This program may be distributed and/or modified under the conditions of the L^AT_EX Project Public License, either version 1.2 of this license or (at your option) any later version. The latest version of this license is in

<http://www.latex-project.org/lppl.txt>

and version 1.2 or later is part of all distributions of L^AT_EX version 1999/12/01 or later.

This program consists of the files `scalebar.dtx` and `scalebar.ins`

Acknowledgements

Thanks to the people on the `comp.text.tex` newsgroup for their help to me in developing this package.

Change History

v1.0
 General: Initial version 1

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.

Symbols	E	I
<code>\,</code> 66	<code>\equal</code> 23	<code>\ifthenelse</code> . . 23, 34, 40
		<code>\isodd</code> 34, 40
C	F	M
<code>\counters/lengths</code> . . <u>4</u>	<code>\FPifint</code> 49, 51, 53	<code>\mandatory_largs</code> <u>1</u>
	<code>\FPset</code> 57	
D	<code>\FPupn</code> 47,	N
<code>\def</code> 44–46, 48, 50, 52, 54	50, 52, 54, 60, 64	<code>\not</code> 33, 39, 59

R			
<code>\raisebox</code>	14, 16, 18, 31, 43, 56	<code>\SB@TextWidth</code>	11, 61-63, 66
S		<code>\SB@Thick</code>	7, 14, 16, 17, 27, 31, 43
<code>\SB@DivisionWidth</code>	10, 13, 14, 17, 18, 30, 37	<code>\SB@unitB</code>	<u>15</u> , 34, 40
<code>\SB@Height</code>	6, 13, 14, 16, 18, 26, 31, 43, 56	<code>\SB@unitT</code>	<u>12</u> , 34, 40
<code>\SB@Length</code>	1, 20, 28, 55	<code>\SBIncrement</code>	47, 49, 50, 64
<code>\SB@MajorWidth</code>	8, 28, 29, 37, 63, 66	<code>\SBMaxNo</code>	45, 47, 53, 54
<code>\SB@MinorWidth</code>	9, 29, 30	<code>\SBMinNo</code>	44, 47, 51, 52, 57
		<code>\SBNextNo</code>	57, 60, 61, 63, 64
		<code>\SBRound</code>	48, 50, 52, 54, 60
		<code>\SBTextUnits</code>	46, 66
		<code>\scalebar</code>	<u>1</u> , <u>19</u>
		<code>\settowidth</code>	61
		T	
		<code>\theSB@countup</code>	33, 39, 59
		<code>\theSB@majordivs</code>	28, 39, 59
		<code>\theSB@minordivs</code>	29, 33
		V	
		<code>\value</code>	34, 40
		W	
		<code>\whiledo</code>	33, 39, 59
