The **tikzsymbols** package*

Ben Vitecek  
b.vitecek@gmx.at  
GitHub

February 8, 2019

Abstract

Some symbols created using tikz.  
For differences between the releases see section 2.  
English is (still) not my native language so there (still) might be some  
errors\(^1\).

Contents

1 Introduction  
2 Important changes  
3 Options

3.1 Load-time Options  
3.1.1 marvosym (true/false)  
3.1.2 prefix (⟨string⟩)  
3.2 Normal Options  
3.2.1 draft (true/false)  
3.2.2 final (true/false)  
3.2.3 tree (true/false/on/off)  
3.2.4 after-symbol (⟨string or command⟩)  
3.2.5 global-scale (⟨number⟩)  
3.2.6 append-style (⟨tikz’ keyval⟩)  
3.2.7 usebox (true/false)  
3.2.8 baseline (true/false)  
3.2.9 remember-picture (true/false)

\(^1\)This document corresponds to **tikzsymbols** v4.10c, dated 2019/02/08.  
\(^1\)They are – of course – on purpose (expect for “available” (sic!)).
1 Introduction

As far as I can remember this package is the result of me writing a cooking book\(^2\). Back then I wasn’t able to find the cooking symbols I wanted and using time, tikz, lot’s of magic (also known as “programming”, but only if the respective person knows what’s going on) and a documentation in bad grammar\(^3\) I somehow ended up with this package.

During time \LaTeX{}3 became known to me and I started experimenting and programming in this (I would say due to its simplicity compared to \LaTeX{}2\(\varepsilon\) far superior) language. Well, long story short: I was impressed. And so the idea of writing my package in \LaTeX{}3 was born.

I finally took my time and started rewriting my code using \LaTeX{}3. This process can be summarized as: “What \textit{does} this command?”, “Why did I define this command?” and more generally “\textit{What} have I done?!”. Well, let’s hope my code (and grammar) is better this time\(^4\).

Well ... that’s it, have fun!

\(^1\)Well, it’s one result, the other one is a cooking book.
\(^2\)Not that it’ now any better.
\(^4\)Looking at own risk. You have been warned.
2 Important changes

The package should behave the same way as the “old” \LaTeX\ 2e release.

2018 Option \texttt{draft} and \texttt{final} are now local.

2017 Option \texttt{usebox} can be used during the document.

\texttt{old} The horribly named command \texttt{\tkzsymbolsaftersymbolinput} is not defined anymore by this package. Please use the new option \texttt{after-symbol}, in combination with the new command \texttt{\tkzsymbolsset}, see section 3 for more information.

\texttt{very old} The option \texttt{draft=absolute} is now obsolete and replaced by the much simpler option \texttt{draft=true}.

3 Options

Options can either be set as package options or using \texttt{\tkzsymbolsset}. Some options can only be set as package options, those are described in section 3.1.

It is recommended to use the option \texttt{draft=true} while working on the document.

\begin{verbatim}
\tkzsymbolsset {
  ⟨keys = values⟩
}\end{verbatim}

Most keys, except for the load-time options (section 3.1), can be set using this command.

3.1 Load-time Options

The following options \textit{cannot} be set using \texttt{\tkzsymbolsset}.

\subsection*{3.1.1 marvosym (true/false)}

\texttt{marvosym = true / false}

Please load \texttt{tikzsymbols} \textit{after} \texttt{marvosym}.

\texttt{marvosym} also defines \texttt{\Smiley} and \texttt{\Coffeecup}. If you prefer those symbols (©, ☕) over the \texttt{tikzsymbols} ones (®, ☕) you can use this option. If set to true \texttt{tikzsymbols} cancels the definition of its \texttt{\Smiley} and \texttt{\Coffeecup}:

\begin{verbatim}
Without option “marvosym”: © ☕ With option “marvosym”: ® ☕
\usepackage{marvosym} \usepackage[marvosym]{tikzsymbols}
\usepackage{tikzsymbols}
\end{verbatim}

This option raises an error if set \texttt{true} without loading package \texttt{marvosym}.

Can only be set as load-time option.

You may also use the option \texttt{prefix} (section 3.1.2).
3.1.2 prefix \((\text{string})\)

This option takes a string as value: \texttt{prefix=\langle string\rangle} and adds this prefix to every command defined by this package. So setting \texttt{prefix=\langle prefix\rangle} adds \langle prefix\rangle to all commands of this package: \texttt{\langle prefix\rangle\textcommand}.

\langle prefix\rangle should neither contain any special characters (e.g., ä, ü, ß, etc.) nor spaces.

By default it is empty, so no prefix is given, if this option is given without an argument \langle prefix\rangle is set to \texttt{tikzsymbols}.

Can only be set as a load-time option.

For example:
\begin{verbatim}
\usepackage[prefix=tikzsym]{tikzsymbols}
\end{verbatim}

defines \texttt{\Smiley} as \texttt{\tikzsymSmiley}, \texttt{\Kochtopf} as \texttt{\tikzsymKochtopf}, \texttt{\pot} as \texttt{\tikzsympot}, etc.

If you use this option or think about using this option the following command may be handy:
\begin{verbatim}
\tikzsymbolsuse\langle Symbolname\rangle
\end{verbatim}

This command takes the name of the symbol without backslash and prints the symbol (or raises an error if the symbol is not defined). Using this command you don’t have to worry about a \langle prefix\rangle, just write the command name and this command adds automatically the given prefix to the command name.

Examples: \tikzsymbolsuse\langle Smiley\rangle[2]
\begin{verbatim}
\tikzsymbolsuse\langle BasicTree\rangle[1.2]\{black\}{red!50!black}\{red\}\{leaf\}
\tikzsymbolsuse\langle Ofen\rangle
\tikzsymbolsuse\langle Fire\rangle[-1.3]
\end{verbatim}

etc.

3.2 Normal Options

Most of these options can be set either as a package-option or with \texttt{\tikzsymbolsset}.

3.2.1 draft \((true/false)\)

\begin{verbatim}
draft draft = \langle true/false\rangle
\end{verbatim}

While working on the document it is recommended to set this option to \texttt{true} because creating many symbols may takes some time to compile and by setting this option to \texttt{true} the symbols are replaced by plain vanilla rectangles (with approximately the same height and width as the symbols) which are faster to create.

You can also set this option during the document.

The old option \texttt{draft=absolute} is obsolete and should therefore not be used.
3.2.2 final (true/false)

\begin{verbatim}
final final= ⟨true/false⟩
\end{verbatim}

This key has the opposite behavior of the option draft. It is a boolean key and therefore accepts only true or false and is set to true by default. Setting it to true prints all symbols normally. Setting it to false prints plain vanilla draft-boxes instead which speeds up the compile-process.

3.2.3 tree (true/false/on/off)

\begin{verbatim}
tree tree= ⟨true/on/false/off⟩
\end{verbatim}

This key accepts true, false and furthermore on and off (for historical reasons). The latter do exactly the same as the first ones.

This option has only an effect on the command \BasicTree and its derivates (\Springtree, \Summertree, \Autumntree and \Wintertree) and substitutes them with tikz drawn boxes.

So while draft=true replaces the output of all commands with simple black boxes, tree=true/on only replaces the output of “tree”-commands with boxes.

It is recommended to use draft=true, but if you want you can use this option.

3.2.4 after-symbol (⟨string or command⟩)

\begin{verbatim}
after-symbol after-symbol = ⟨⟨string or command⟩⟩
\end{verbatim}

Is more stable if set using \tikzsymbolsset. The value of this key is inserted after every command of this package. By default it is set to \xspace.
3.2.5 global-scale \((\textit{number})\)
symbol-scale \((\langle\textit{key-value list}\rangle)\)

\begin{verbatim}
global-scale = \{(number)\}
symbol-scale = \{\langle\textit{symbol-i=number-1, symbol-2=number-2,...}\rangle\}
\end{verbatim}

global-scale can be used to scale \textit{all} commands by given \(\langle\textit{number}\rangle\).

If only some specific symbols should be scaled, you may use the second option and specify which symbol or symbols (name of the symbol without backslash) should be scaled. Using the german name (if available) has the same effect as using the english one.

\textbf{Note:} You can scale the symbols in this package in three different ways: The first is to scale \textit{all} symbols using \texttt{global-scale}, the second is scaling specific symbols using \texttt{symbol-scale} and the third is by using the optional argument provided by the symbols (which I call \texttt{local-scale}; e.g. \texttt{\textbackslash Smiley[2]}).

The important thing is that those scaling methods \textit{do not cancel} each other, but behave multiplicative.

If a local scale is given (e.g. \texttt{\textbackslash Smiley[2]}) with \texttt{global-scale=3} the resulting scaling will be \(3 \cdot 2 = 6\). Is furthermore this specific symbol is also scaled (e.g. by 1.1), the resulting scaling (for this symbol) will be \(3 \cdot 1.1 \cdot 2 = 6.6\).

\textbf{Examples:} \texttt{\textbackslash tikzsymbolset\{symbol-scale=\{ Smiley= 1.5 \\}}\

\begin{verbatim}
\texttt{\textbackslash tikzsymbolset\{symbol-scale=\{ Smiley= 1.5 \}\}} \\
\texttt{\textbackslash tikzsymbolset\{symbol-scale=\{ Smiley= 5 \}\}} \\
\texttt{\textbackslash tikzsymbolset\{symbol-scale=\{ Smiley= 2, Schneebesen=2.1 \}\}} \\
\texttt{\textbackslash tikzsymbolset\{global-scale=3,symbol-scale=\{ Smiley= 2, Schneebesen=2.1 \}\}}
\end{verbatim}

\textbf{Note:} Using “eggbeater” instead of “Schneebesen” does the same thing.

3.2.6 append-style \((\langle\texttt{tikz’ keyval}\rangle)\)

\begin{verbatim}
append-style = \{(\texttt{tikz’ keyval})\}
\end{verbatim}

With this option you can append \texttt{\textbackslash tikz’ \langle keyval\rangle} to \texttt{tikzsymbol} internal style.

\textbf{Note:} The style is called \_\_\texttt{tikzsymbol} and while the name will probably not change, you are discouraged to use it directly unless it is \textit{really} necessary (e.g. if I did something wrong).
3.2.7 usebox (true/false)

usebox = \{true/false\}
In tikzsymbols all symbols are stored inside boxes (\sbox) and while I still have no idea what exactly happens, it shortens the compilation time of the document. By default this option is true.

The drawback is that \LaTeX\ has only a limited number of box registers. If you come across an error message regarding boxes try setting usebox=false.

3.2.8 baseline (true/false)

baseline = \{true/false\}
This option mainly exists to let the commands of this package work inside \texttt{todonotes}' \texttt{\todo} command. If set to true it adds to each symbol of this package the \texttt{tikz} option \texttt{baseline=default}. If you do not want this, set this option to false. It is set to true by default.

3.2.9 remember-picture (true/false)

remember-picture = \{true/false\}

Adds to each symbol created by this package the \texttt{tikz} option \texttt{remember picture=(true/false)}. It is not added by default.

4 Symbols

In this section the symbols are introduced. They automatically change automatically with text-size. 

\begin{tikzsymbols}
\end{tikzsymbols}
## 4.1 Cooking-symbols

The following table shows all available cooking-symbols and their respective commands. The first column shows the command-names (german & english), the second the optional parameter(s). The optional parameter(s) are for both the german and the english commands the same.

(scale) can be a number between (not exactly) −1400 and (also not exactly) 1400, default is 1.

Da Umlaute nicht in Befehlsnamen vorkommen dürfen, werden die Umlaute ö, ä, ü durch o, a, u ersetzt.

<table>
<thead>
<tr>
<th>German &amp; English Commands</th>
<th>Optional parameter(s)</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>\Kochtopf</td>
<td>\pot</td>
<td>![image]</td>
</tr>
<tr>
<td>\Bratpfanne</td>
<td>\fryingpan</td>
<td>![image]</td>
</tr>
<tr>
<td>\Schneebesen</td>
<td>\eggbeater</td>
<td>![image]</td>
</tr>
<tr>
<td>\Sieb</td>
<td>\sieve</td>
<td>![image]</td>
</tr>
<tr>
<td>\Purierstab</td>
<td>\blender</td>
<td>![image]</td>
</tr>
<tr>
<td>\Dreizack</td>
<td>\trident</td>
<td>![image]</td>
</tr>
<tr>
<td>\Backblech</td>
<td>\bakingplate</td>
<td>![image]</td>
</tr>
<tr>
<td>\Ofen</td>
<td>\oven</td>
<td>![image]</td>
</tr>
<tr>
<td>\Pfanne</td>
<td>\pan</td>
<td>![image]</td>
</tr>
<tr>
<td>\Herd</td>
<td>\cooker</td>
<td>![image]</td>
</tr>
<tr>
<td>\Saftpresse</td>
<td>\squeezer</td>
<td>![image]</td>
</tr>
<tr>
<td>\Schussel</td>
<td>\bowl</td>
<td>![image]</td>
</tr>
<tr>
<td>\Reibe</td>
<td>\peeler</td>
<td>![image]</td>
</tr>
<tr>
<td>\grater</td>
<td>\grater</td>
<td>![image]</td>
</tr>
<tr>
<td>\Flasche</td>
<td>\bottle</td>
<td>![image]</td>
</tr>
<tr>
<td>\Nudelholz</td>
<td>\rollingpin</td>
<td>![image]</td>
</tr>
<tr>
<td>\Knoblauchpresse</td>
<td>\garlicpress</td>
<td>![image]</td>
</tr>
</tbody>
</table>
### 4.2 Emoticons 😊

#### 4.2.1 “Normal” Emoticons 😊

First column shows the commands, the second the (optional) parameter(s), the third the default-output (the only command with a mandatory argument is \Changey).

\( \langle \text{scale} \rangle \) can be a number between (not exactly) \(-2000\) and (not exactly) \(2000\), default is \(1\).

\( \langle \text{color} \rangle \) can be every defined color. Note: The color names shouldn’t contain special characters like ß, ä, ö, ...

\Changey’s \( \langle \text{mood} \rangle \) has to be between \(-2\) and \(2\) (\(1\) equals \Smiley, \(-1\) \Sadey and \(0\) \Neutrey).

\SchrodingersCat’s \( \langle \text{case} \rangle \) can either be \(1\) (alive), \(0\) (unknown) or \(-1\) (dead).

<table>
<thead>
<tr>
<th>Commands</th>
<th>(Optional) parameter(s)</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>\Smiley</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )]</td>
<td>😊</td>
</tr>
<tr>
<td>\Sadey</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )]</td>
<td>😊</td>
</tr>
<tr>
<td>\Neutrey</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )]</td>
<td>😊</td>
</tr>
<tr>
<td>\Changey</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )] {(\langle \text{mood} \rangle )}</td>
<td>😊</td>
</tr>
<tr>
<td>\cChangey</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color1} \rangle )[(\langle \text{color2} \rangle )[(\langle \text{color3} \rangle )] {(\langle \text{mood} \rangle )}</td>
<td>😊</td>
</tr>
<tr>
<td>\Annoey</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )]</td>
<td>😊</td>
</tr>
<tr>
<td>\Laughey</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )] {(\langle \text{mouth color} \rangle )}</td>
<td>😊</td>
</tr>
<tr>
<td>\Winkey</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )]</td>
<td>😊</td>
</tr>
<tr>
<td>\oldWinkey</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )]</td>
<td>😊</td>
</tr>
<tr>
<td>\Sey</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )]</td>
<td>😊</td>
</tr>
<tr>
<td>\Xey</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )]</td>
<td>😊</td>
</tr>
<tr>
<td>\Innocey</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )] {(\langle \text{halo color} \rangle )}</td>
<td>😊</td>
</tr>
<tr>
<td>\WInnocey</td>
<td>( \langle \text{scale} \rangle )</td>
<td>😊</td>
</tr>
<tr>
<td>\Cooley</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )]</td>
<td>😊</td>
</tr>
<tr>
<td>\Tongey</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )] {(\langle \text{tongue color} \rangle )}</td>
<td>😊</td>
</tr>
<tr>
<td>\Nursey</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )] {(\langle \text{cap color} \rangle )[(\langle \text{cross color} \rangle )]}</td>
<td>😊</td>
</tr>
<tr>
<td>\Vomey</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )] {(\langle \text{vomit color} \rangle )}</td>
<td>😊</td>
</tr>
<tr>
<td>\Walley</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )] {(\langle \text{wall color} \rangle )}</td>
<td>😊</td>
</tr>
<tr>
<td>\rWalley</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )] {(\langle \text{wall color} \rangle )}</td>
<td>😊</td>
</tr>
<tr>
<td>\Cat</td>
<td>( \langle \text{scale} \rangle )</td>
<td>😊</td>
</tr>
<tr>
<td>\SchrodingersCat</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{case} \rangle )]</td>
<td>😎</td>
</tr>
<tr>
<td>\Ninja</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )] {(\langle \text{headband color} \rangle )[(\langle \text{eye color} \rangle )]</td>
<td>😎</td>
</tr>
<tr>
<td>\Sleepey</td>
<td>( \langle \text{scale} \rangle )[(\langle \text{color} \rangle )] {(\langle \text{cap color} \rangle )[(\langle \text{star color} \rangle )]</td>
<td>😊</td>
</tr>
<tr>
<td>\NiceReapey</td>
<td>( \langle \text{scale} \rangle )</td>
<td>😊</td>
</tr>
</tbody>
</table>

“r” for “random generated cracks”.

Examples: \Sadey[] [red] 😎
If you intend to change the color of \textcolor{blue} you may define a new command so that you do not have to write those brackets each time.
4.2.2 “3D” Emoticons 😊😊

First column shows the commands (note: the “3D” Emoticons begin with \d...), the second shows the (optional) parameter(s), the third shows the default-output (the only command with a mandatory argument is \dChangey).

⟨scale⟩ can be a number between a small number (under −500 for sure) and a large number (over 500 for sure), default is 1.

⟨color⟩ can be every defined color (see examples below). Note: The color names shouldn’t contain special characters like ß, ä, ö, ...

\dChangey’s ⟨mood⟩ has to be between −2 and 2 (1 equals \dSmiley, −1 \dSadey and 0 \dNeutrey).

<table>
<thead>
<tr>
<th>Commands</th>
<th>Optional parameter(s)</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>\dSmiley</td>
<td>[(scale)][(color)]</td>
<td>😊</td>
</tr>
<tr>
<td>\dSadey</td>
<td>[(scale)][(color)]</td>
<td>😊</td>
</tr>
<tr>
<td>\dNeutrey</td>
<td>[(scale)][(color)]</td>
<td>😊</td>
</tr>
<tr>
<td>\dcChangey</td>
<td>[(scale)][(color)][{mood}]</td>
<td>😊</td>
</tr>
<tr>
<td>\dChangey</td>
<td>[(scale)][(color)][{(color1)][(color2)][(color3)]{mood}</td>
<td>😊</td>
</tr>
<tr>
<td>\dAnnoey</td>
<td>[(scale)][(color)]</td>
<td>😊</td>
</tr>
<tr>
<td>\dLaughey</td>
<td>[(scale)][(color)][{mouth color}]</td>
<td>😊</td>
</tr>
<tr>
<td>\dSey</td>
<td>[(scale)][(color)]</td>
<td>😊</td>
</tr>
<tr>
<td>\dKey</td>
<td>[(scale)][(color)]</td>
<td>😊</td>
</tr>
<tr>
<td>\dInnocey</td>
<td>[(scale)][(color)][{halo color}]</td>
<td>😊</td>
</tr>
<tr>
<td>\dCooley</td>
<td>[(scale)][(color)]</td>
<td>😊</td>
</tr>
<tr>
<td>\dTongey</td>
<td>[(scale)][(color)][{tongue color}]</td>
<td>😊</td>
</tr>
<tr>
<td>\dNursey</td>
<td>[(scale)][(color)][{cap color}][{cross color}]</td>
<td>😊</td>
</tr>
<tr>
<td>\dVomey</td>
<td>[(scale)][(color)][{vomit color}]</td>
<td>😊</td>
</tr>
<tr>
<td>\dWinkey</td>
<td>[(scale)][(color)][{wall color}]</td>
<td>😊</td>
</tr>
<tr>
<td>\drWalley</td>
<td>[(scale)][(color)][{wall color}]</td>
<td>😊</td>
</tr>
<tr>
<td>\dNinja</td>
<td>[(scale)][(color)][{headband color}][{eye color}]</td>
<td>😊</td>
</tr>
<tr>
<td>\dlddWinkey</td>
<td>[(scale)][(color)][{cap color}][{star color}]</td>
<td>😊</td>
</tr>
</tbody>
</table>

“r” for “random generated cracks”.

Examples: \dSadey[][red] 😊
\dCooley[-3][cyan] 😊
\dVomey[1.5][green!70!black][olive] 😊
\dNursey[][yellow][blue][red] 😊
\dNinja[1.3][][violet][red] 😊
\dChangey{-2} 😊 \dChangey{-1.367} 😊 \dChangey{-1} 😊 \dChangey{0} 😊 \dChangey{1} 😊 \dChangey{1.41} 😊 \dChangey{2} 😊
If you intent to change the color of \texttt{dcChangey} you may define a new command so that you do not have to write those brackets each time.

### 4.3 Other Symbols

\texttt{\textbackslash Strichmaxerl}'s optional parameters 2–5 (\texttt{\langle left arm\rangle} to \texttt{\langle right leg\rangle}) can be a number between $-360$ and $360$ (of course the number can be even greater or even smaller.). The parameters are the angles between the body and the separate parts of \texttt{\textbackslash Strichmaxerl} (see examples).

\texttt{\langle scale\rangle} can be a very great and a very small negative number (but I don’t think, that you need so large symbols).

\texttt{\langle color\rangle} can be every defined color. Note: The color names shouldn’t contain special characters like ß, ä, ö, ….

<table>
<thead>
<tr>
<th>Commands</th>
<th>Optional parameter(s)</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{Strichmaxerl}</td>
<td>\texttt{\langle scale\rangle} \texttt{\langle left arm\rangle} \texttt{\langle right arm\rangle} \texttt{\langle left leg\rangle} \texttt{\langle right leg\rangle}</td>
<td><img src="image" alt="" /></td>
</tr>
<tr>
<td>\texttt{Candle}</td>
<td>\texttt{\langle scale\rangle}</td>
<td><img src="image" alt="" /></td>
</tr>
<tr>
<td>\texttt{Fire}</td>
<td>\texttt{\langle scale\rangle}</td>
<td><img src="image" alt="" /></td>
</tr>
<tr>
<td>\texttt{Coffeecup}</td>
<td>\texttt{\langle scale\rangle}</td>
<td><img src="image" alt="" /></td>
</tr>
<tr>
<td>\texttt{Chair}</td>
<td>\texttt{\langle scale\rangle}</td>
<td><img src="image" alt="" /></td>
</tr>
<tr>
<td>\texttt{Bed}</td>
<td>\texttt{\langle scale\rangle}</td>
<td><img src="image" alt="" /></td>
</tr>
<tr>
<td>\texttt{Moai}</td>
<td>\texttt{\langle scale\rangle}</td>
<td><img src="image" alt="" /></td>
</tr>
<tr>
<td>\texttt{Tribar}</td>
<td>\texttt{\langle scale\rangle} \texttt{\langle color 1\rangle} \texttt{\langle color 2\rangle} \texttt{\langle color 3\rangle}</td>
<td><img src="image" alt="" /></td>
</tr>
<tr>
<td>\texttt{Snowman}</td>
<td>\texttt{\langle scale\rangle}</td>
<td><img src="image" alt="" /></td>
</tr>
</tbody>
</table>

\texttt{Tribar[-10]} \texttt{\langle color\rangle} [red] [green]

\texttt{Tribar[2.1]} \texttt{\langle color\rangle} [blue] [blue!50] [blue!20]

\texttt{Strichmaxerl[1][10][30][40][4]} ![](image), \texttt{Strichmaxerl[1.4][210][310][10][90]} ![](image), \texttt{Strichmaxerl[2][510][110][190][990]} ![](image), \texttt{Strichmaxerl[0.9][54][28][95][16]} ![](image), \texttt{Strichmaxerl[]} ![](image)
### 4.4 Trees 🌳

(scale) can be a number between (not exactly) $-900$ and (again not exactly) $900$, default is 1.

(color) can be every defined color (see examples below). Note: The color names shouldn’t contain special characters like ß, ä, ö, ....

{⟨leaf⟩} uses the colors of {⟨leaf color a⟩} and {⟨leaf color b⟩}, you can leave this one empty if you don’t want leaves (∉Wintertree is without leaf, see examples below).

If you are using those trees, \LaTeX{} needs longer to produce the output. So you may use the package option \texttt{tree=off}, or (better) \texttt{draft=true} (see section section 3.2.1 and section 3.2.3) to make \LaTeX{} faster.

Furthermore those trees are pretty much stolen from the tikz manual.

<table>
<thead>
<tr>
<th>Commands</th>
<th>Optional/Needed parameter(s)</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>\BasicTree</td>
<td>[⟨scale⟩]⟨{trunk color}⟩⟨{leaf color a}⟩⟨{leaf color b}⟩⟨{leaf}⟩</td>
<td>see below</td>
</tr>
<tr>
<td>\Springtree</td>
<td>[⟨scale⟩]</td>
<td>🌳</td>
</tr>
<tr>
<td>\Summertree</td>
<td>[⟨scale⟩]</td>
<td>🌳</td>
</tr>
<tr>
<td>\Autumntree</td>
<td>[⟨scale⟩]</td>
<td>🌳</td>
</tr>
<tr>
<td>\Wintertree</td>
<td>[⟨scale⟩]</td>
<td>🌳</td>
</tr>
<tr>
<td>\WorstTree</td>
<td>[⟨scale⟩]</td>
<td>🌳</td>
</tr>
</tbody>
</table>

\BasicTree examples Some “normal” trees:

\begin{verbatim}
\colorbox{green}{\BasicTree{red}{orange}{yellow}{leaf}}
\colorbox{black}{\BasicTree{red}{orange}{yellow}{leaf}}
\end{verbatim}

\begin{verbatim}
\BasicTree[5]{orange!95!black}{orange!80!black}{orange!70!black}{leaf}
\BasicTree[-1.54]{green!20!black}{green!50!black}{green!70!black}{leaf}
\end{verbatim}

\begin{verbatim}
\BasicTree[3.75]{gray!80}{gray!50}{gray!40}{leaf}
\end{verbatim}

\begin{draftbox}\BasicTree examples Some “draftbox” trees (using \texttt{tree=false}):

\begin{verbatim}
\colorbox{green}{\BasicTree{red}{orange}{yellow}{leaf}}
\end{verbatim}

\begin{verbatim}
\BasicTree[2]{blue!65!white}{cyan!50!white}{cyan!50!white}\{}
\BasicTree[-1.54]{green!20!black}{green!50!black}{green!70!black}{leaf}
\end{verbatim}

\begin{verbatim}
\colorbox{black}{\BasicTree[3.75]{gray!80}{gray!50}{gray!40}{leaf}}
\end{verbatim}

\end{draftbox}
5 FAQ (Known errors and problems)

Or “Questions I assume would be frequently asked, if people would frequently ask questions”.

5.1 How to get rid of the space after each symbol?

By default the package adds \xspace after each command. To remove it use the option after-symbol. Using

\tikzsymbolsset{after-symbol={}}

removes the \xspace command and thus the unwanted space.

5.2 Using the symbols causes unwanted ⟨problem⟩. How could I get rid of it?

This could have something to do with question 5.5 (after you made sure that the symbols cause the problem). Try using setting the option usebox=false and recompile a few times. If the problem persists, please send a bug report (section 6).

5.3 I am getting the error-message Argument of \pgffor@next has an extra }

If you encounter an error message like

Argument of \pgffor@next has an extra }
while using \texttt{babel} with e.g. language “français” and for example \texttt{\LaTeX} you may add

\begin{verbatim}
\usetikzlibrary{babel}
\end{verbatim}

to your preamble. This should (hopefully) fix the problem.

5.4 Another package I load already defines \textit{(symbol)}.

You can override pretty much every symbol simply by loading \texttt{tikzsymbols} last as it defines the symbols via \texttt{\DeclareDocumentCommand} (see \texttt{xparse}).

If you want to use the symbols of both packages you may have a look at option \texttt{prefix}.

5.5 Does this package store symbols in boxes and reuses them instead of creating a new picture every time?

Yes, it does. It can become a problem if \LaTeX{} runs out of boxes. If this happens, use \texttt{usebox=false}.

Furthermore, \texttt{tikz} allows to reference pictures using e.g. \texttt{remember picture}. This also influences the symbols of \texttt{tikzsymbols}. As those symbols are stored and copied for printing, labels attached to the symbols get repeated. In this case, also try using \texttt{usebox=false} (or try the option \texttt{remember-picture=false}).

5.6 Are the symbols created with the environment \texttt{tikzpicture}?

Yes, they are.

6 Nobody is perfect

If you find a bug please send me a mail (or report it on GitHub) involving a \textit{minimal example} showing the bug and a short description (english or german). Please mention (if you are writing a mail) “\texttt{tikzsymbols}” in the header, “\texttt{gmx}” has a habit of putting mails into the spam-folder and it helps me to recognize those mails faster. This can also be the reason why I may need some time to answer the mail.

As I am also new to GitHub, I also may take longer to answer, at least until I figured out how to get a mail if a new issue is created.

Suggestions are also welcome.

7 Danksagung

I would like to thank all users for providing bug reports and helping to improve this package.

Furthermore many thanks to my brother helping me improving the symbols.
8 Changes

See the “README.md” file.