tikzsymbols*

Ben Vitecek
b.vitecek@gmx.at

October 31, 2014

Abstract

Just some symbols created using tikz.
English is not my native language. So there (still) might be some errors ☺

Contents

*This document corresponds to tikzsymbols v3.0f, dated 2014/10/31.

1 Short Introduction

There are about two emoticons available in LaTeX: Smiley and Frowny. But why aren't there more? Or why did nobody make cooking-symbols[1]? I thought about this questions and during a project I developed some (cooking)symbols. Developing them was real fun and so I made some more, reworked them etc. And here they are.

2 Options

2.1 tree=on/true/off/false, draft=true/false

These options are only relevant for the commands in the section "Trees" (3.4). The trees look pretty nice (at least I think they do), but have one drawback: LaTeX needs extremely long to produce them. So these options come in handy: by setting `tree=off/false` or using `draft=true` or simply `draft` the trees will be replaced by squares drawn by tikz (for examples see section "Trees" 3.4).

One drawback is that these options only change the `\BasicTree` command, but not the others. Also another drawback is that tikz is still used to draw, this means LaTeX will be slowed down if you use many symbols. You may use `draft=absolute`, which is – I think – more useful.

2.2 draft=absolute

Use this option if you use many symbols!

This option replaces *all* symbols by fast drawn plain vanilla rectangles, which have (mostly) the exactly same proportions as the tikzsymbols. For example, by setting `draft=absolute \Smiley` will produce □, `\Nursey` □, `\BasicTree{red}{red!50!black}{black}{leaf}`□, `\Schussel`□, etc.

You see, they are *very* plain and *very* vanilla (but LaTeX needs no time to produce them).[2]

2.3 final=true/false

This option is the opposite of `draft=true/false`.

[1] Well, there are some, but not the one I wanted.
[2] The old option `draftabsolute` is still useable, but obsolete.

2.4 marvosym=true/false

You can use this special option if you also use package marvosym. If you want the marvosym Smiley (☺) and Coffeecup (☕) instead of the tikzsymbols ones (☺, ☕) you can use option `marvosym` resp. `marvosym=true`. If you use this option, tikzsymbols will simply not define its Smiley and Coffeecup.

Note: *Always load tikzsymbols after marvosym.*

Without option "marvosym": ☺ ☕	With option "marvosym": ☺ ☕
`\usepackage{marvosym}` `\usepackage{tikzsymbols}`	`\usepackage{marvosym}` `\usepackage{marvosym}{tikzsymbols}`

If you use option `marvosym` without loading the package marvosym, LaTeX will produce an error message.

This option is *false* by default.

2.5 usebox=true/false

Since v3.0 the symbols are stored inside a `\savebox`[3] and are used with `\usebox`. The advantage is that LaTeX doesn't need to recalculate the symbol again[4] (if you use the *exactly* same symbol, see section 3 for more information). This option is `true` by default.

You can deactivate this storing-inside-saveboxes by setting `usebox=false`.

To activate it you can use `usebox=true` or just `usebox`.

2.6 prefix, prefix=⟨*prefix*⟩

If you use a package which collides with tikzsymbols, but want to use symbols of both packages, you can use this option. It adds a ⟨*prefix*⟩ to all symbol commands provided by tikzsymbols. All commands will look like `\<prefix>command`, for example: `\<prefix>Smiley`, `\<prefix>drWalley`, `\<prefix>Springtree`, etc.

If you simply use option `prefix`, `<prefix>` will be "tikzsymbols": `\Smiley` will change to `\tikzsymbolsSmiley`, `\drWalley` to `\tikzsymbolsdrWalley`, `\tikzsymbolsSpringtree`, `\tikzsymbolsBasicTree`, etc.

If this prefix is too long for you, you can define your own prefix via `prefix=<prefix>`. `<prefix>` should neither contain any special characters (e.g., ä, ü, ß, etc.) nor empty spaces. For example (using `prefix=T`): `\Smiley` changes to `\TSmiley`, `\Kochtopf` to `\TKochtopf` (and `\pot` to `\Tpot`), etc.

`\tikzsymbolsuse` If you change the `prefix` often or are not sure if you will change it in future, you may work with `\tikzsymbolsuse{}` to use the symbols without worrying about the prefix. `\tikzsymbolsuse{}` takes one mandatory argument: the command-name of the symbol *without* backslash. Write the optional and mandatory parameters of the symbol after the curly braces.

[3] Inside a `\sbox` to be correctly.

[4] I think.

For example: \tikzsymbolsuse{Smiley}[2]
\tikzsymbolsuse{BasicTree}[1.2]{black}{red!50!black}{red}{leaf}
\tikzsymbolsuse{Ofen} \tikzsymbolsuse{Fire}[-1.3]
etc.

3 Symbols

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

Furthermore since v3.0 this package uses a savebox-usebox system. That means the output of a symbol is saved inside a box[5] using \sbox and every time you use the *exactly* same symbol, LaTeX just can use the already calculated symbol (via \usebox).

What is "the *exactly* same symbol"? Using a symbol with the same optional parameter(s), the same script size and text-color.

For example: ☺ and ☺ and ☺ were only calculated once because they have the same optional parameter, script size and text-color.

☹, ☹ and ☹ would be calculated twice: the first time for the black one , the second time for the red one due to having a different text-color. The third Sadey is the same as the first, so it doesn't have to be recalculated.

😐, 😐, 😐 and 😐 have all to be calculated separately: the first time for the normal, black one, the second time for the blue one (due to text color), the third time for having a different script size than the first one and the fourth time due to having a different script size and a different text-color.

Each symbol is stored in a separate box, but I think using etex with 32768 box registers is enough (and I don't think that you are using so many symbols; if you managed somehow to use so many boxes you can deactivate the storing of tikzsymbols-symbols inside boxes using usebox=false (see 2.5)).

There is a great advantage using this box-system: once calculated, the symbol can be used again without any new calculation[6] (I know I am repeating myself). This means, you can use for example \Summertree many times without having to wait till LaTeX finished recalculating all of them (again, only if they are *exactly* the same): . With version < 3.0 or option usebox=false this would take *extremely* long.

And again I will give you an advice: If you are using trees or many symbols, you should use option draft=absolute.

3.1 cooking-symbols

At the following table the cooking-symbols are listed.

[5] To say the truth: I am (still) not sure what exactly happens while storing commands inside \sbox and using them via \usebox...

[6] And hopefully no disadvantage...

The first column shows the commands (at first the german at second the english ones). In the second the optional parameter(s) are shown. 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[7] , default is 1.

Da Umlaute nicht angezeigt werden können, werden die Umlaute ö, ä, ü durch: o, a, u ersetzt.

German & English Commands		Optional parameter(s)	Output
`\Kochtopf`	`\pot`	[⟨*scale*⟩]	
`\Bratpfanne`	`\fryingpan`	[⟨*scale*⟩]	
`\Schneebesen`	`\eggbeater`	[⟨*scale*⟩]	
`\Sieb`	`\sieve`	[⟨*scale*⟩]	
`\Purierstab`	`\blender`[8]	[⟨*scale*⟩]	
`\Dreizack`	`\trident`	[⟨*scale*⟩]	
`\Backblech`	`\bakingplate`	[⟨*scale*⟩]	
`\Ofen`	`\oven`	[⟨*scale*⟩]	
`\Pfanne`	`\pan`	[⟨*scale*⟩]	
`\Herd`	`\cooker`	[⟨*scale*⟩]	
`\Saftpresse`	`\squeezer`	[⟨*scale*⟩]	
`\Schussel`	`\bowl`	[⟨*scale*⟩]	
`\Schaler`	`\peeler`	[⟨*scale*⟩]	
`\Reibe`	`\grater`	[⟨*scale*⟩]	
`\Flasche`	`\bottle`	[⟨*scale*⟩]	

3.2 Emoticons ☺

3.2.1 "normal" Emoticons ☠

First column shows the commands, the second the optional parameter(s), the third the default-output.

⟨*scale*⟩ can be a number between (not exactly) −2000 and (not exactly) 2000[9], default is 1.

⟨*color*⟩ can be every defined color. Note: The color names shouldn't contain special characters like ß, ä, ö, ...

Commands	Optional parameter(s)	Output
`\Sadey`	[⟨*scale*⟩] [⟨*color*⟩]	☹
`\Smiley`	[⟨*scale*⟩] [⟨*color*⟩]	☺
`\Laughey`	[⟨*scale*⟩] [⟨*color*⟩] [⟨*mouth color*⟩]	😄
`\Annoey`	[⟨*scale*⟩] [⟨*color*⟩]	😑

[7] Since version 2.2 you can use negative numbers as well (see examples)

[8] I know that "Pürierstab" should be translated as "immersion blender", but I'm just using "blender"

[9] Do you even need so large symbols?

Commands	Optional parameter(s)	Output
\Neutrey	[⟨*scale*⟩] [⟨*color*⟩]	
\Winkey	[⟨*scale*⟩] [⟨*color*⟩]	
\oldWinkey	[⟨*scale*⟩] [⟨*color*⟩]	
\Sey	[⟨*scale*⟩] [⟨*color*⟩]	
\Xey	[⟨*scale*⟩] [⟨*color*⟩]	
\Innocey	[⟨*scale*⟩] [⟨*color*⟩] [⟨*halo color*⟩]	
\wInnocey	[⟨*scale*⟩]	
\Cooley	[⟨*scale*⟩] [⟨*color*⟩]	
\Tongey	[⟨*scale*⟩] [⟨*color*⟩] [⟨*tongue color*⟩]	
\Nursey	[⟨*scale*⟩] [⟨*color*⟩] [⟨*cap color*⟩] [⟨*cross color*⟩]	
\Vomey	[⟨*scale*⟩] [⟨*color*⟩] [⟨*vomit color*⟩]	
\Walley	[⟨*scale*⟩] [⟨*color*⟩] [⟨*wall color*⟩]	
\rWalley[10]	[⟨*scale*⟩] [⟨*color*⟩] [⟨*wall color*⟩]	
\Cat	[⟨*scale*⟩]	
\Ninja	[⟨*scale*⟩] [⟨*color*⟩] [⟨*headband color*⟩] [⟨*eye color*⟩]	
\NiceReapey	[⟨*scale*⟩]	

Examples: \Sadey[][red] \Cooley[-3][cyan]
\Vomey[1.5][green!80!black][olive].
\Nursey[][yellow][blue][red].
\Ninja[1.3][][violet][red].
\colorbox{yellow}{\Winkey \Annoey[-1]\Neutrey}
{\color{blue}\Sey}

3.2.2 "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.

⟨*scale*⟩ can be a number between a small number[11] and a large number[12], 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, ...

Commands	Optional parameter(s)	Output
\dSadey	[⟨*scale*⟩] [⟨*color*⟩]	
\dSmiley	[⟨*scale*⟩] [⟨*color*⟩]	
\dLaughey	[⟨*scale*⟩] [⟨*color*⟩] [⟨*mouth color*⟩]	
\dAnnoey	[⟨*scale*⟩] [⟨*color*⟩]	
\dNeutrey	[⟨*scale*⟩] [⟨*color*⟩]	

[10] "r" stands for "random" and means that the cracks in the wall are generated randomly, but it takes some time to generate them.
[11] under −500 for sure
[12] over 500 for sure

Commands	Optional parameter(s)	Output
`\dWinkey`	[⟨*scale*⟩] [⟨*color*⟩]	
`\olddWinkey`	[⟨*scale*⟩] [⟨*color*⟩]	
`\dSey`	[⟨*scale*⟩] [⟨*color*⟩]	
`\dXey`	[⟨*scale*⟩] [⟨*color*⟩]	
`\dInnocey`	[⟨*scale*⟩] [⟨*color*⟩] [⟨*halo color*⟩]	
`\dCooley`	[⟨*scale*⟩] [⟨*color*⟩]	
`\dTongey`	[⟨*scale*⟩] [⟨*color*⟩] [⟨*tongue color*⟩]	
`\dNursey`	[⟨*scale*⟩] [⟨*color*⟩] [⟨*cap color*⟩] [⟨*cross color*⟩]	
`\dVomey`	[⟨*scale*⟩] [⟨*color*⟩] [⟨*vomit color*⟩]	
`\dWalley`	[⟨*scale*⟩] [⟨*color*⟩] [⟨*wall color*⟩]	
`\drWalley`[13]	[⟨*scale*⟩] [⟨*color*⟩] [⟨*wall color*⟩]	
`\dNinja`	[⟨*scale*⟩] [⟨*color*⟩] [⟨*headband color*⟩] [⟨*eye color*⟩]	

Examples: `\dSadey[][red]` `\dCooley[-3][cyan]`
`\dVomey[1.5][green!70!black][olive]`.
`\dNursey[][yellow][blue][red]`.
`\dNinja[1.3][][violet][red]`.

3.3 other Symbol(s)

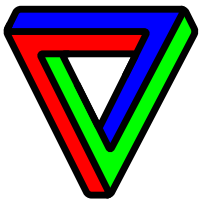
`\Strichmaxerl`'s optional parameters 2–5 (⟨*left arm*⟩ to ⟨*right leg*⟩) can be a number between -360 and 360[14]. These parameters are the angles between the body and the separate parts of `\Strichmaxerl` (see examples).

⟨*scale*⟩ can be a very great and a very small number (but I don't think, that you need so large symbols).

⟨*color*⟩ can be every defined color. Note: The color names shouldn't contain special characters like ß, ä, ö, ...

Commands	Optional parameter(s)	Output
`\Strichmaxerl`	[⟨*scale*⟩] [⟨*left arm*⟩] [⟨*right arm*⟩] [⟨*left leg*⟩] [⟨*right leg*⟩]	
`\Candle`	[⟨*scale*⟩]	
`\Fire`	[⟨*scale*⟩]	
`\Coffeecup`	[⟨*scale*⟩]	
`\Chair`	[⟨*scale*⟩]	
`\Bed`	[⟨*scale*⟩]	
`\Moai`	[⟨*scale*⟩]	
`\Tribar`	[⟨*scale*⟩] [⟨*color 1*⟩] [⟨*color 2*⟩] [⟨*color 3*⟩]	
`\Snowman`	[⟨*scale*⟩]	

[13] "r" stands for "random" and means that the cracks in the wall are generated randomly, but it takes some time.
[14] Of course the number can be even greater or less, but it doesn't make sense.



`\Tribar[-10][blue][red][green]`

`\Tribar[2.1][blue][blue!50][blue!20]`

`\Strichmaxerl[1][10][30][40][4]`, `\Strichmaxerl[1.4][210][310][10][90]`  ,
`\Strichmaxerl[2][510][110][190][990]`, `\Strichmaxerl[0.9][54][28][95][16]` 

3.4 Trees 

"Hey, these trees look exactly like the ones in the tikzmanual" – "NO! Not 'exactly', they look pretty a like... Well I changed them a bit... ... Hey! The best ideas are stolen ..."

⟨*scale*⟩ can be a number between (not exactly) −900 and (again not exactly) 900[15], 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 `tree=off`, `draft` or (better) `draft=absolute` (see section 2) to make LaTeX faster.

Commands	Optional/Needed parameter(s)	Output
`\BasicTree`	`[`⟨*scale*⟩`]{`⟨*trunk color*⟩`}{`⟨*leaf color a*⟩`}{`⟨*leaf color b*⟩`}{`*leaf*`}`	see below
`\Springtree`	`[`⟨*scale*⟩`]`	
`\Summertree`	`[`⟨*scale*⟩`]`	
`\Autumntree`	`[`⟨*scale*⟩`]`	
`\Wintertree`	`[`⟨*scale*⟩`]`	
`\WorstTree`	`[`⟨*scale*⟩`]`	

`\BasicTree` examples (normal trees):

`\colorbox{green}{\BasicTree{red}{orange}{yellow}{leaf}}` 

`\BasicTree[5]{orange!95!black}{orange!80!black}{orange!70!black}{leaf}`

`\BasicTree[2]{blue!65!white}{cyan!50!white}{cyan!50!white}{}` 

`\BasicTree[-1.54]{green!20!black}{green!50!black}{green!70!black}{leaf}` 

`\colorbox{black}{\BasicTree[3.75]{gray!80}{gray!50}{gray!40}{leaf}}`

[15] if it is larger (or less) it uses too much of LaTeX memory and an error message appears.

. . . and using the same trees with `tree=off/false` or `draft(=true)`:

```
\colorbox{green}{\BasicTree{red}{orange}{yellow}{leaf}}
\BasicTree[5]{orange!95!black}{orange!80!black}{orange!70!black}{leaf}
\BasicTree[2]{blue!65!white}{cyan!50!white}{cyan!50!white}{}
\BasicTree[-1.54]{green!20!black}{green!50!black}{green!70!black}{leaf}
\colorbox{black}{\BasicTree[3.75]{gray!80}{gray!50}{gray!40}{leaf}}
```

I think it's better if you define your own tree using `\newcommand` and `\BasicTree`:

```
\newcommand{\Myicetree}[1][1]{%
\BasicTree[#1]{blue!65!white}{cyan!50!white}{cyan!50!white}{}}
```

3.5 Something to redefine

At the end of each symbol `\tikzsymbolsaftersymbolinput` is inserted. By default it is defined to contain `\xspace`:

```
\newcommand{\tikzsymbolsaftersymbolinput}{\xspace}
```

You can redefine this macro. If you don't want `\xspace` just say:

```
\renewcommand{\tikzsymbolsaftersymbolinput}{}
```

4 Warnings and Errors

4.1 Warnings . . .

You can use this symbols in chapters, sections, subsections, etc. But the log file will print a warning, something like:

> Package hyperref Warning: Token not allowed in a PDF string (PDF-DocEncoding): (hyperref) removing '`\Smiley`' on input line 137.

You can avoid those messages by putting the symbol into this command:
`\texorpdfstring{\Smiley}{Smiley}`
For example you may use something like that:

```
\subsubsection{\enquote{3D} Emoticons \texorpdfstring{\dSmiley}{dSmiley}}
```

or

```
\subsection{Emoticons \texorpdfstring{\Smiley}{Smiley}}
```

or

```
\subsubsection{\enquote{normal} Emoticons \texorpdfstring{\Cat}{Cat}}
```

4.2 . . . and errors

Make sure you load marvosym *before* tikzsymbols because both packages define `\Smiley`, marvosym via `\newcommand` tikzsymbols via `\DeclareRobustCommand`.

If you load marvosym *after* tikzsymbols, LaTeX generates an error-message because `\Smiley` has already been defined.

If you load marvosym *before* tikzsymbols, tikzsymbols will overwrite marvosym's Smiley and no error-message is generated (if you like the `\Smiley` from marvosym more, use the tikzsymbols option `marvosym` or `prefix`).

5 Nobody is perfect

If you find a bug, please send me a mail involving a *minimal example* which shows the bug. And a description would be nice.

6 Code (do you really need this section?)

There is not much to see, all this symbols were created with tikz. But it may helps you (somehow).

The first lines are always the same: what do I need, how is the package named:

```
\NeedsTeXFormat{LaTeX2e}[2011/06/27]
\ProvidesPackage{tikzsymbols}
[2014/10/31 v3.0f Some symbols created using tikz.]
\@ifpackageloaded{tikz}{}{\RequirePackage{tikz}}
\@ifpackageloaded{xargs}{}{\RequirePackage{xargs}}
\@ifpackageloaded{xcolor}{}{\RequirePackage{xcolor}}
\@ifpackageloaded{xkeyval}{}{\RequirePackage{xkeyval}}
\@ifpackageloaded{xspace}{}{\RequirePackage{xspace}}
\@ifpackageloaded{calc}{}{\RequirePackage{calc}}
```

Furthermore we need to load some libraries from tikz:

```
\usetikzlibrary{arrows,decorations.pathmorphing,trees}
```

`\if@tkzssmbls@neg` `\iftikzymbols@draftabsolute`

We need `\if@tkzssmbls@neg` (=negative), well . . . if something is negative (`\chair` needs this).

`\iftikzymbols@draftabsolute` is needed for option `draft=absolute`.

```
\newif\if@tkzssmbls@neg
\newif\iftikzymbols@draftabsolute
```

`\Basic@Tree`

Now we define our `\Basic@Tree`. We will need it later for our package option. Furthermore if no option is given this will be the default definition of `\Basic@Tree` inside the document.

```
\newcommand{\Basic@Tree}{\Basic@Tree@on}
```

`final`

If final is false, `\Basic@Tree` will show squares drawn by tikz. If it is true, it will show trees.

```
\define@boolkey{tikzsymbols}{final}[true]{%
  \ifKV@tikzsymbols@final
    \def\Basic@Tree{\Basic@Tree@on}% final=true => final
  \else
    \def\Basic@Tree{\Basic@Tree@off}% = false  => draft
  \fi
}
```

`draft` If option `draft` or `draft=true` is set, then squares drawn by tikz are typed instead of trees.

If option `draft=absolute` is set, we set `\iftikzymbols@draftabsolute` to `true` and *all* symbols are replaced by plain vanilla rectangles drawn by LaTeX.

```
\define@choicekey*{tikzsymbols}{draft}%
[\tikzsymbols@draft@val\tikzsymbols@draft@nr]{false,true,absolute}[true]{%
  \ifcase\tikzsymbols@draft@nr\relax
    \def\Basic@Tree{\Basic@Tree@on}% draft=false => final
  \or
    \def\Basic@Tree{\Basic@Tree@off}% = true  => draft
  \or
    \def\Basic@Tree{\Basic@Tree@off}\tikzymbols@draftabsolutetrue% =absolute
  \fi
}
```

`draftabsolute` Obsolete Option `draftabsolute`. You can still use it, but it gives a warning.

```
\define@key{tikzsymbols}{draftabsolute}{%
\typeout{-----------------------------}%
\PackageWarningNoLine{tikzsymbols}{Option '\CurrentOption' is obsolete!
\MessageBreak Please use 'draft=absolute' instead!}
\typeout{-----------------------------}%
\tikzymbols@draftabsolutetrue}
```

`marvosym` Now I define the boolean option `marvosym`: you may use it, if you load the package marvosym.

```
\define@boolkey{tikzsymbols}{marvosym}[true]{}
```

`usebox` If it is true, the savebox-usebox-system is used. If it is false, then the system is turned off.

Furthermore it is true by default (`\KV@tikzsymbols@useboxtrue`)

```
\define@boolkey{tikzsymbols}{usebox}[true]{}
\KV@tikzsymbols@useboxtrue
```

`prefix` `\cmdKV@tikzsymbols@prefix` Option `prefix` changes all commands to `\<prefix>command`. If only `prefix` is set, `<prefix>` will be "tikzsymbols", but you can define your own prefix via `prefix=<prefix>`

If this option is not used inside a document, `\cmdKV@tikzsymbols@prefix` wouldn't be defined. So we define it and let it empty.

```
\newcommand{\cmdKV@tikzsymbols@prefix}{}
\define@cmdkey{tikzsymbols}{prefix}[tikzsymbols]{}
```

`tree` Now we declare the name of our option: `tree` (I could have named it stone, or wood, etc., but I used "tree"). I am using xkeyval now more than before and `tree` can now be set to `on/true` resp. `off/false`.

```
\define@choicekey{tikzsymbols}{tree}%
[\tikzsymbols@tree@val\tikzsymbols@tree@nr]{true,on,false,off}[on]{%
  \ifcase\tikzsymbols@tree@nr\relax
     \def\Basic@Tree{\Basic@Tree@on}% tree=true
   \or
     \def\Basic@Tree{\Basic@Tree@on}% tree=on
   \or
     \def\Basic@Tree{\Basic@Tree@off}% tree=false
   \or
     \def\Basic@Tree{\Basic@Tree@off}% tree=false
  \fi
}
```

`\ProcessOptionsX*` We process all options. What is `\relax` doing?

```
\ProcessOptionsX*<tikzsymbols>\relax
```

`\tikzsymbolsaftersymbolinput` Now we define this strange named macro. This macro is inserted after the tikz-code, and is defined as `\xspace`. If you don't want `\xspace` to be inserted, you can redefine this command.

```
\newcommand*{\tikzsymbolsaftersymbolinput}{\xspace}
```

`\@leaf@is@leaf` We need this command for creating an error-message if the last parameter of BasicTree is neither "leaf" nor empty.

```
\newcommand*{\@leaf@is@leaf}{leaf}
```

`\tkzsymbls@scl` The `[`⟨*scale*⟩`]` parameter of the commands is stored inside `\tkzsymbls@scl`.

`\set@tkzsymblsscl` is a short version of `\setlength{\tkzsymbls@scl}{#1pt}`. I defined it to write less.

```
\newlength{\tkzsymbls@scl}
\newcommand*{\set@tkzsymblsscl}[1]{\setlength{\tkzsymbls@scl}{#1pt}}
```

`\tikzsymbols@draftboxlength` `\tikzsymbols@draftboxheight` The length and the height of the plain vanilla rectangle are stored inside these lengths.

```
\newlength{\tikzsymbols@draftboxlength}
\newlength{\tikzsymbols@draftboxheight}
```

`\tikzsymbols@draftbox` Our plain vanilla rectangle. Using `\tikzsymbols@draftboxlength` and `\tikzsymbols@draftboxheight` to calculate the length and the height of the box.

The box is drawn using `\frame` and a `\vbox`, `\hbox` construction. `\vbadness=\maxdimen` is needed because otherwise there would be many overfull v-box errors.

```
\newcommand*{\tikzsymbols@draftbox}[2]{%
  \setlength{\tikzsymbols@draftboxlength}{#1}%
  \setlength{\tikzsymbols@draftboxheight}{#2}%
  \frame{%
    {\vbadness=\maxdimen%
```

```
    \frame{\vbox to \tikzsymbols@draftboxheight{%
      \hbox to \tikzsymbols@draftboxlength{}}}%
  }%
}
```

`\tikzsymbols@draftQbox` Short form of `\tikzsymbols@draftbox`. "Q" means "Quadrat"(*square*) and just means that height and length of the box are the same.

```
\newcommand*{\tikzsymbols@draftQbox}[1]{\tikzsymbols@draftbox{#1}{#1}}
```

`\tkzsymbls@Prmtr` `\tksymbls@bx@Prmtrstore` `\tikzsymbols@draftbox` doesn't like negative numbers. So to be sure that only positive numbers are used, `\tksymbls@bx@Prmtrstore` changes `\tkzsymbls@Prmtr` to be positive (if it is negative). I am using `\tkzsymbls@Prmtr` instead of `#1` (scaling) in `\tikzsymbols@draftbox`

```
\newcommand*\tkzsymbls@Prmtr{}
\newcommand*\tksymbls@bx@Prmtrstore[1]{%
  \edef\tkzsymbls@Prmtr{\ifdim\tkzsymbls@scl<0pt-\fi#1}%
}
```

`\current@tikzsymbols` The `\sbox` and `\usebox` system I am using now has one drawback: If the symbol was used inside e.g. `\color{red}` all other symbols of the same type are red, even if they are in normal text, or even inside another color.

The same with e.g. `\small`. If a symbol was used the first time inside e.g. `\small` all other symbols of the same type are as small as the first one.

To overcome this problem, the save-boxes name depends of the current color and the current script size. These things are stored inside `\current@tikzsymbols`.

```
\newcommand*{\current@tikzsymbols}{}
```

`\tikzsymbols@ifsaveboxundefined` For every symbol we define a box (using `\sbox`). If a symbol of the same type is used again, it doesn't have to be recalculated. We simply use the same output as the first symbol. If the symbol has another input, a new save-box is defined.

For example: `\Summertree[1]\Summertree[1]\Summertree[1]` is only calculated once because they are alike:  . `\Summertree[1.3]` will be saved in an own save-box because it is different then the others.

```
\newcommand*{\tikzsymbols@ifsaveboxundefined}[2]{%
  \edef\current@tikzsymbols{\current@color\f@size}%
  \expandafter\ifcsname\current@tikzsymbols tikzsymbols@savebox@#1\endcsname%
  \relax%
  \else%
  \expandafter\expandafter\expandafter\newsavebox%
  \expandafter\expandafter\expandafter{%
    \csname\current@tikzsymbols tikzsymbols@savebox@#1\endcsname}%
  \expandafter\expandafter\expandafter\global%
  \expandafter\expandafter\expandafter\sbox%
  \expandafter\expandafter\expandafter{%
    \csname\current@tikzsymbols tikzsymbols@savebox@#1\endcsname}{#2}%
  \fi%
}
```

`\tikzsymbols@use@box` tikzsymbols version of `\usebox`.

```
\newcommand*\tikzsymbols@use@box[1]{%
  \expandafter\expandafter\expandafter\usebox%
  \expandafter\expandafter\expandafter{%
    \csname\current@tikzsymbols tikzsymbols@savebox@#1\endcsname}%
}
```

`\ifKV@tikzsymbols@usebox = true` If option `usebox` is `false`, then `\tikzsymbols@ifsaveboxundefined` and `\tikzsymbols@use@box` will be redefined so that they don't store the input inside a save-box.

```
\ifKV@tikzsymbols@usebox
  \relax% if true
  \else
    \renewcommand{\tikzsymbols@ifsaveboxundefined}[2]{\@secondoftwo{#1}{#2}}%
    \renewcommand{\tikzsymbols@use@box}[1]{\@gobble{#1}}%
\fi
```

`kzsymbls@Declare@Robust@Command` To make use of the `prefix` option (and to write less), I am using these commands. If `<prefix>` is empty, the commands will be "normal": `\Smiley` will be `\Smiley`, etc.

If `<prefix>` is not empty the commands will be defined as `\<prefix>command` e.g. `\<prefix>Smiley` , etc.

Furthermore a new command is defined. `tikzsymbolsuse` needs this command to specify wherever the input is a symbol of tikzsymbols or not.

```
\newcommand{\tkzsymbls@Declare@Robust@Command}[1]{%
  \expandafter\def\csname\cmdKV@tikzsymbols@prefix#1tikzsymbolsuse\endcsname{}%
  \expandafter\DeclareRobustCommand%
  \csname\cmdKV@tikzsymbols@prefix#1\endcsname%
}
```

`zsymbls@Declare@Robust@Commandx` Same as before

```
\newcommand{\tkzsymbls@Declare@Robust@Commandx}[1]{%
  \expandafter\def\csname\cmdKV@tikzsymbols@prefix#1tikzsymbolsuse\endcsname{}%
  \expandafter\DeclareRobustCommandx%
  \csname\cmdKV@tikzsymbols@prefix#1\endcsname%
}
```

`\tkzsymblsnewcommand` Same as before

```
\newcommand{\tkzsymblsnewcommand}[1]{%
  \expandafter\def\csname\cmdKV@tikzsymbols@prefix#1tikzsymbolsuse\endcsname{}%
  \expandafter\newcommand%
  \csname\cmdKV@tikzsymbols@prefix#1\endcsname%
}
```

`\tikzsymbolsuse` To be able to don't have to care about the prefix, I made this command. Simply write the name of the symbols inside without backslash. If the symbol is not defined, there will be an error message.

```
\newcommand{\tikzsymbolsuse}[1]{%
  \ifcsname\cmdKV@tikzsymbols@prefix#1tikzsymbolsuse\endcsname%
```

```
  \relax\else\PackageError{tikzsymbols}{\MessageBreak%
    Undefined Control sequence: '#1'}{Did you write the name correctly?}\fi%
  \csname\cmdKV@tikzsymbols@prefix#1\endcsname%
}
```

\tikzsymbols@let

\eggbeater is the same as \Schneebesen (\let\eggbeater\Schneebesen). To make the prefix also for the \let commands, I defined my own let-command

```
\newcommand{\tikzsymbols@let}[2]{%
  \expandafter\def\csname\cmdKV@tikzsymbols@prefix#1tikzsymbolsuse\endcsname{}%
  \expandafter\expandafter\expandafter\let%
  \expandafter\csname\expandafter\cmdKV@tikzsymbols@prefix%
  \expandafter#1\expandafter\endcsname%
  \csname\cmdKV@tikzsymbols@prefix#2\endcsname%
}
```

\@Tree@SetUp

First we define our \@Tree@SetUp (how the trees will look like) (I used the code from the tikz manual and changed it a little bit):

```
\def\@Tree@SetUp{\tikzset{%
    ld/.style={level distance=##1ex},lw/.style={line width=##1ex},%
    level 1/.style={ld=0.60,    trunk,               lw=0.1 ,sibling angle=60},%
    level 2/.style={ld=0.20,    trunk!80!leaf a,     lw=.073,sibling angle=70},%
    level 3/.style={ld=0.25,    trunk!60!leaf a,     lw=.05,sibling angle=70}, %
    level 4/.style={ld=0.10,    trunk!40!leaf a,     lw=.025,sibling angle=60},%
    level 5/.style={ld=0.15,    trunk!20!leaf a,     lw=.02,sibling angle=60},%
    level 6/.style={ld=0.08,    leaf a,              lw=.021,sibling angle=60},%
}}%
```

\Basic@Tree

\Basic@Tree@off and \Basic@Tree@on are necessary for option tree, draft and final.

\Basic@Tree@ff

\Basic@Tree@off is used when trees are turned off.

```
\DeclareRobustCommandx{\Basic@Tree@off}[5][1=1, usedefault]{{%
    \set@tkzsymblsscl{#1}%
    \pgfmathsetmacro\tikzsymbols@Tree@absolute@scale{#1+0.02ex}%
    \edef\tkzsymbls@Prmtr{\ifdim\tkzsymbls@scl<0pt-\fi\tikzsymbols@Tree@absolute@scale}%
    \ifdim\tkzsymbls@scl<0pt \set@tkzsymblsscl{-#1}\fi%
    \def\leaf@or@not@leaf{#5}%
    \iftikzymbols@draftabsolute%
      \ifx\leaf@or@not@leaf\@leaf@is@leaf%
        \tikzsymbols@draftbox{(1.6772ex+0.4pt)*\real{\tkzsymbls@Prmtr}}
        {(1.42ex-0.2pt+0.4pt)*\real{\tkzsymbls@Prmtr}}%
      \else
        \tikzsymbols@draftbox{(1.3996ex+0.4pt)*\real{\tkzsymbls@Prmtr}}
        {(1.28ex-0.2pt+0.4pt)*\real{\tkzsymbls@Prmtr}}%
      \fi
    \else%
      \begin{tikzpicture}[scale=#1+0.02ex,x=1ex,y=1ex, line width=0.4pt*\tkzsymbls@scl]
        \ifx\leaf@or@not@leaf\@leaf@is@leaf%
          \draw[#2] (-0.8386,0+0.2pt) -- (-0.8386,1.42);
```

```
          \draw[#3] (-0.8386,1.42) -- (0.8386,1.42);
          \draw[#4] (0.8386,1.42) -- (0.8386,0+0.2pt);
          \draw[#3] (0.8386,0+0.2pt) -- (0,0+0.2pt);
          \draw[#4] (0,0+0.2pt) -- (-0.8386,0+0.2pt);
        \else
          \draw[#2] (-0.6998,0+0.2pt) -- (-0.6998,0.68+0.6);
          \draw[#3] (-0.6998,0.68+0.6) -- (0.6998,0.68+0.6);
          \draw[#4] (0.6998,0.68+0.6) -- (0.6998,0+0.2pt);
        \fi%
      \end{tikzpicture}%
    \fi%
}}
```

\Basic@Tree@on \Basic@Tree@on is used when trees are turned on.

```
\DeclareRobustCommandx{\Basic@Tree@on}[5][1=1, usedefault]{{%
  \iftikzymbols@draftabsolute%
    \Basic@Tree@off[#1]{#2}{#3}{#4}{#5}%
  \else%
    \set@tkzsymblsscl{#1}%
    \ifdim\tkzsymbls@scl<0pt \set@tkzsymblsscl{-#1}\@tkzssmbls@negtrue\fi%
    \def\leaf@or@not@leaf{#5}%
    \@Tree@SetUp%
    \colorlet{trunk}{#2}%
    \colorlet{leaf a}{#3}%
    \colorlet{leaf b}{#4}%
    \begin{tikzpicture}[x=1ex,y=1ex,line width=0.07ex]%
      \pgfarrowsdeclare{leaf}{leaf}%
      {\pgfarrowsleftextend{-.1ex}\pgfarrowsrightextend{-0.05ex}}%
      {%
        \pgfpathmoveto{\pgfpoint{-.01ex}{0ex}}%
        \pgfpatharc{150}{30}{0.08ex}%
        \pgfpatharc{-30}{-150}{0.08ex}%
        \pgfusepathqfill%
      }%
      \ifx\leaf@or@not@leaf\@leaf@is@leaf%
      \draw[opacity=0,scale=#1+0.02ex, line width=0.4pt*\tkzsymbls@scl]
      (-0.8386,0+0.2pt) rectangle
      (0.8386, 1.42);
      \else %
      \draw[opacity=0,scale=#1+0.02ex, line width=0.4pt*\tkzsymbls@scl]
      (-0.6998,0+0.2pt) rectangle (0.6998,0.68+0.6);
      \fi
      \pgflowlevel{\pgftransformscale{#1+0.02ex}}{%
        \coordinate (root) [grow cyclic,rotate=90] child {
          child [line cap=round] foreach \a in {0,1, 2} { child foreach \b in {0,1} {
              child foreach \c in {0,1,2} { child foreach \d in {0,1} {
                  child foreach \leafcolor in {leaf a,leaf b} { edge from parent [color=\leafco
                }}}} edge from parent [shorten >=-0.05ex,serif cm-,line cap=butt]
      };}%
\end{tikzpicture}%
```

```
 \@tkzssmbls@negfalse%
 \fi%
}}
```

6.1 Cookingsymbolcode

\Kochtopf = \pot

I am using \DefineRobustCommand so that the symbols can be used inside \section{}, \footnote, \index{}, etc. It may would have worked with \newcommand too.

You can either use the german commands or the english ones:

```
\tkzsymbls@Declare@Robust@Command{Kochtopf}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Kochtopf#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{2.47ex*\real{\tkzsymbls@Prmtr}}{1.577ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2ex,y=2.2ex, line width=0.07ex*\tkzsymbls@scl,scale=#1]
\draw[rounded corners=0.2ex*\tkzsymbls@scl] (0,0.5) -- (0,0) -- (1,0) -- (1,0.5);
\draw(0,0.4) arc (90:270:0.1);
\draw(1,0.4) arc (90:-90:0.1);
\draw (0,0.5) -- (1,0.5) .. controls (1,0.6) and (0,0.6) .. (0,0.5);
\draw (0.6,0.585) arc (0:180:0.1);
\draw[decorate,
decoration={snake,amplitude=.12ex*\tkzsymbls@scl,segment length=0.93ex*\tkzsymbls@scl}]
(0,0.35) -- (1,0.35);
\draw (0.1,0.25) circle (0.04);
\draw (0.3,0.2) circle (0.04);
\draw (0.13, 0.125) circle (0.04);
\draw (0.6,0.25) circle (0.04);
\draw (0.45,0.1) circle (0.04);
\draw (0.88,0.2) circle (0.04);
\draw (0.7,0.11) circle (0.04);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Kochtopf#1}%
\tikzsymbolsaftersymbolinput%
}
\tikzsymbols@let{pot}{Kochtopf}
```

\Bratpfanne = \fryingpan

If you wonder why I am using `line width=0.07ex*\tkzsymbls@scl` instead of `line width=0.07ex*#1` I will try to explain it.

After being multiplied by a negative number, the line widths would be too thin for the size of the symbol. So it is necessary that the line width is always scaled with a positive number. Thus I am using \tkzsymbls@scl because it is always positive.

```
\tkzsymbls@Declare@Robust@Command{Bratpfanne}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Bratpfanne#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{3.5535ex*\real{\tkzsymbls@Prmtr}}{1.4525ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=0.7ex,y=1.4ex, line width=0.07ex*\tkzsymbls@scl, scale=#1,
decoration={snake,amplitude=.05ex*\tkzsymbls@scl,segment length=0.408ex*\tkzsymbls@scl}]
\draw[rounded corners=0.07ex*\tkzsymbls@scl]
(-1,0) -- (1,0) -- (1.5,0.4) -- (-1.5,0.4) -- cycle;
\draw[ line width=0.037ex*\tkzsymbls@scl, rounded corners=0.023ex*\tkzsymbls@scl]
(-1.4,0.3) -- (-3.5,0.3) -- (-3.5,0.25) -- (-1.3,0.25);
\draw[line width=0.023ex*\tkzsymbls@scl] (-1.1,0.1) -- (1.1,0.1);
\draw[line width=0.035ex*\tkzsymbls@scl, decorate]
(-0.3,0.5) -- (-0.3,1);
\draw[line width=0.035ex*\tkzsymbls@scl, decorate]
   (0.3,0.5) -- (0.3,1);
\draw[line width=0.035ex*\tkzsymbls@scl, decorate]
(-1,0.5) -- (-1,1);
\draw[line width=0.035ex*\tkzsymbls@scl, decorate]
   (1,0.5) -- (1,1);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Bratpfanne#1}%
\tikzsymbolsaftersymbolinput%
}
\tikzsymbols@let{fryingpan}{Bratpfanne}
```

`\Schneebesen = \eggbeater` The next one:

```
\tkzsymbls@Declare@Robust@Command{Schneebesen}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Schneebesen#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{0.5697ex*\real{\tkzsymbls@Prmtr}}{1.57985ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[y=2.1ex,x=1.4ex, scale=#1,
   line width=0.01ex*\tkzsymbls@scl*\real{0.97}]
\draw (0,0) .. controls (0.2,0.0) and  (0.2,0.2) ..  (0,0.4);
\draw (0,0) .. controls (-0.2,0.0) and  (-0.2,0.2) ..  (0,0.4);
\draw  (0,0) .. controls (-0.1,0.0) and  (-0.1,0.2) ..  (0,0.4);
\draw (0,0) .. controls (0.1,0.0) and  (0.1,0.2) ..  (0,0.4);
\draw (0,0) .. controls (-0.15,0.0) and  (-0.15,0.2) ..  (0,0.4);
\draw (0,0) .. controls (0.15,0.0) and  (0.15,0.2) ..  (0,0.4);
\draw (0,0) .. controls (-0.05,0.0) and  (-0.05,0.2) ..  (0,0.4);
\draw  (0,0) .. controls (0.05,0.0) and  (0.05,0.2) ..  (0,0.4);
```

```
\draw (0,0) --(0,0.4);
\fill[line width=0.05ex*\tkzsymbls@scl, rounded corners=0.07ex*\tkzsymbls@scl]
   (-0.05,0.37) -- (0.05,0.37) -- (0.05,0.75) -- (-0.05,0.75) -- cycle;
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Schneebesen#1}%
\tikzsymbolsaftersymbolinput%
}
\tikzsymbols@let{eggbeater}{Schneebesen}
```

`\Sieb = \sieve` Now a long one;

```
\tkzsymbls@Declare@Robust@Command{Sieb}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Sieb#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{3.478ex*\real{\tkzsymbls@Prmtr}}{1.175ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.8ex, y=2.8ex,line width=0.02ex*\tkzsymbls@scl ,scale=#1]
\draw[line width=0.09ex*\tkzsymbls@scl] (-0.2,0) -- (1.01,0);
\draw (0.2,0) arc (180:360:0.4);
\draw(0.25,0) arc (180:360:0.35);
\draw (0.3,0) arc (180:360:0.3);
\draw (0.35,0) arc (180:360:0.25);
\draw (0.4,0) arc (180:360:0.2);
\draw (0.45,0) arc (180:360:0.15);
\draw (0.5,0) arc (180:360:0.1);
\draw (0.55,0) arc (180:360:0.05);
\draw (.95,0) -- (0.95,-0.194);
\draw (.9,0) -- (0.9,-0.265);
\draw (.85,0) -- (0.85,-0.313);
\draw (.8,0) -- (0.8,-0.345);
\draw (.75,0) -- (0.75,-0.37);
\draw (.7,0) -- (0.7,-0.39);
\draw (.65,0) -- (0.65,-0.4);
\draw (.6,0) -- (0.6,-0.4);
\draw (.55,0) -- (0.55,-0.4);
\draw (.5,0) -- (0.5,-0.39);
\draw (.45,0) -- (0.45,-0.37);
\draw (.4,0) -- (0.4,-0.348);
\draw (.35,0) -- (0.35,-0.314);
\draw (.3,0) -- (0.3,-0.265);
\draw (.25,0) -- (0.25,-0.194);
\draw (0.2,-0.05) -- (1,-0.05);
\draw (0.21,-0.1) -- (0.99,-0.1);
\draw (0.23,-0.15) -- (0.97,-0.15);
\draw (0.255,-0.2) -- (0.945,-0.2);
\draw (0.289,-0.25) -- (0.911,-0.25);
```

```
\draw (0.335,-0.3) -- (0.865,-0.3);
\draw (0.406,-0.35) -- (0.794,-0.35);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Sieb#1}%
\tikzsymbolsaftersymbolinput%
}
\tikzsymbols@let{sieve}{Sieb}
```

`\Purierstab = \blender` Da es keine Umlaute gibt, werden ä, ü, ö einfach zu: a, u, o. This symbol is far from perfect. And I know that the correct translation of "Pürierstab" would be "immersion blender", but I am just using "blender":

```
\tkzsymbols@Declare@Robust@Command{Purierstab}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Purierstab#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{0.76ex*\real{\tkzsymbls@Prmtr}}{1.575ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.3ex, y=2.2ex, line width=0.07ex*\tkzsymbls@scl,scale=#1]
\draw[rounded corners=0.07ex*\tkzsymbls@scl] (0,0) -- (0.3,0) --  (0.15,0.1) --cycle;
\fill[rounded corners=0.07ex*\tkzsymbls@scl] (0.15,0.3) -- (0.24,0.4) -- (0.24,0.7) --
   (0.06,0.7) -- (0.06,0.4) -- cycle;
\draw (0.15,0.4) -- (0.15,0.1);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Purierstab#1}%
\tikzsymbolsaftersymbolinput%
}
\tikzsymbols@let{blender}{Purierstab}
```

`\Dreizack = \trident` Important cooking-tool for cooking:

```
\tkzsymbls@Declare@Robust@Command{Dreizack}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Dreizack#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{0.265ex*\real{\tkzsymbls@Prmtr}}{1.575ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.3ex, y=2.2ex, line width=0.035ex*\tkzsymbls@scl,scale=#1]
\fill[rounded corners=0.07ex*(\tkzsymbls@scl-\tkzsymbls@scl/100)]
(0,0) -- (0,0.4)  -- (0.1,0.4) -- (0.1,0.0) -- cycle;
\draw (0.05,0) -- (0.05,0.7);
\draw[rounded corners=0.07ex*(\tkzsymbls@scl-\tkzsymbls@scl/100*\tkzsymbls@scl*2]
(0,0.7) -- (0,0.55)  -- (0.05,0.55) -- (0.1,0.55) --  (0.1,0.7);
```

```
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Dreizack#1}%
\tikzsymbolsaftersymbolinput%
}
\tikzsymbols@let{trident}{Dreizack}
```

`\Backblech = \bakingplate` I may have too many strange named commands:

```
\tkzsymbls@Declare@Robust@Command{Backblech}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Backblech#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{2.3155ex*\real{\tkzsymbls@Prmtr}}{1.57ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=6.53ex,y=5ex, line width=0.07ex*\tkzsymbls@scl,scale=#1]
\filldraw[rounded corners=0.09ex*\tkzsymbls@scl] (0,0) rectangle (0.3,0.3);
\draw[rounded corners=0.07ex*\tkzsymbls@scl, line width=0.03ex*\tkzsymbls@scl]
   (0.1,0) -- (-0.025,0) -- (-0.025,0.3) -- (0.1,0.3);
\draw[rounded corners=0.07ex*\tkzsymbls@scl, line width=0.03ex*\tkzsymbls@scl]
   (0.2,0) -- (.325,0) -- (.325,0.3) -- (0.2,0.3);
\foreach \@BackblechlochX in {0.007,0.293}
\foreach \@BackblechlochY in {0.007,0.293}
\fill[white] (\@BackblechlochX,
\@BackblechlochY) circle (0.02ex);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Backblech#1}%
\tikzsymbolsaftersymbolinput%
}
\tikzsymbols@let{bakingplate}{Backblech}
```

`\Ofen = \oven` I may have again too many strange named commands:

```
\tkzsymbls@Declare@Robust@Command{Ofen}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Ofen#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{2.07ex*\real{\tkzsymbls@Prmtr}}{1.57ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=0.50ex,y=0.5ex, line width=0.07ex*\tkzsymbls@scl,scale=#1]
\draw (0,0) rectangle (4,3);
\draw (0.25,0.25) rectangle (3.75,2);
\foreach \@Ofenschalter in {0.5,1.1,2.9,3.5}
\fill (\@Ofenschalter,2.5) circle (0.22);
\draw (1.5,2.28) rectangle  (2.5,2.72);
```

```
\draw[line width=0.05ex*\tkzsymbls@scl] (1,1.75) -- (3,1.75);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Ofen#1}%
\tikzsymbolsaftersymbolinput%
}
\tikzsymbols@let{oven}{Ofen}
```

`\Pfanne = \pan` A pan ... What did you expect?

```
\tkzsymbls@Declare@Robust@Command{Pfanne}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Pfanne#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{3.034ex*\real{\tkzsymbls@Prmtr}}{0.78ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.3ex,y=2.3ex, line width=0.09ex*\tkzsymbls@scl,scale=#1]
\draw [rounded corners=0.023ex*\tkzsymbls@scl]
   (0,0) -- (0.9,0) -- (1,0.3) -- (-0.1,0.3) -- cycle;
\draw (-0.2,0.22) -- (-0.08,0.22);
\draw (0.97,0.22) -- (1.08,0.22);
\draw[decorate,decoration={snake,amplitude=.046ex*\tkzsymbls@scl,
   segment length=0.82ex*\tkzsymbls@scl},line width=0.05ex*\tkzsymbls@scl]
   (-0.05,0.1) -- (0.95,0.1);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Pfanne#1}%
\tikzsymbolsaftersymbolinput%
}
\tikzsymbols@let{pan}{Pfanne}
```

`\Herd = \cooker` I hope it's the right translation:

```
\tkzsymbls@Declare@Robust@Command{Herd}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Herd#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{2.08ex*\real{\tkzsymbls@Prmtr}}{1.58ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=1ex,y=1ex,line width=0.04ex*\tkzsymbls@scl,scale=#1]
\draw[line width=0.08ex*\tkzsymbls@scl] (0,0) rectangle (2,1.5);
\draw (0.5,0.45) circle (0.35);
\draw (0.5,0.45) circle (0.2);
\draw (1.45,0.45) circle (0.3);
\draw (0.5,1.15) circle (0.21);
\draw (1.05,0.95) rectangle (1.85,1.35);
```

```
\draw (1.45,1.15) circle (0.15);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Herd#1}%
\tikzsymbolsaftersymbolinput%
}
\tikzsymbols@let{cooker}{Herd}
```

`\Saftpresse = \squeezer` It's an old squeezer:

```
\tkzsymbls@Declare@Robust@Command{Saftpresse}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Saftpresse#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{1.87ex*\real{\tkzsymbls@Prmtr}}{1.62ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=1.2ex,y=1ex,line width=0.07ex*\tkzsymbls@scl,scale=#1]
\draw[rounded corners=0.1ex*\tkzsymbls@scl]
   (0,0.85) -- (0,0) -- (1.5,0) -- (1.5,0.85) -- cycle;
\draw (0,0.7) -- (1.5,0.7);
\draw[rounded corners=0.1ex*\tkzsymbls@scl] (0.3,0.7) -- (0.75,1.55) -- (1.2,0.7);
\draw[rounded corners=0.1ex*\tkzsymbls@scl] (0.45,0.7) -- (0.75,1.55) -- (1.05,0.7);
\draw[rounded corners=0.1ex*\tkzsymbls@scl]
   (0.65,0.7) -- (0.75,1.55) -- (0.85,0.7);
\draw[line width=0.05ex*\tkzsymbls@scl, decorate,
   decoration={snake,amplitude=.05ex*\tkzsymbls@scl,
   segment length=0.48ex*\tkzsymbls@scl}]  (0,0.3) -- (1.5,0.3);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Saftpresse#1}%
\tikzsymbolsaftersymbolinput%
}
\tikzsymbols@let{squeezer}{Saftpresse}
```

`\Schussel = \bowl` It may looks a bit weird, but I like it. Wieder dasselbe mit den Umlauten: ü=u.

```
\tkzsymbls@Declare@Robust@Command{Schussel}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Schussel#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{2.32ex*\real{\tkzsymbls@Prmtr}}{1.47ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=1ex,y=1ex,line width=0.07ex*\tkzsymbls@scl, scale=#1]
\draw[rounded corners=0.5ex*\tkzsymbls@scl]
   (-0.02,1.4) -- (0,1.4) -- (0,0.05) -- (1.5,0.05) -- (1.5,1.4) -- (1.52,1.4);
\draw (0.35,0) -- (1.15,0);
```

```
\draw[opacity=00] (-0.4,0) -- (1.85,0);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Schussel#1}%
\tikzsymbolsaftersymbolinput%
}
\tikzsymbols@let{bowl}{Schussel}
```

\Schaler = \peeler I cannot believe I forgot this command. I made it and forgot to copy and paste it inside this document! Jedenfalls wieder ä=a:

```
\tkzsymbls@Declare@Robust@Command{Schaler}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Schaler#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{1.15ex*\real{\tkzsymbls@Prmtr}}{1.565ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.7ex,y=2.3ex, line width=0.07ex*\tkzsymbls@scl,scale=#1]
\draw[rounded corners=0.07ex*\tkzsymbls@scl]
   (0,0.4) -- (0,0.1) arc (0:180:-0.1) -- (0.2,0.4)
   -- (0.3,0.5) -- (0.3,0.65) -- (0.2,0.65) -- (0.2,0.5) -- (0,0.5) -- (0,0.65) --
   (-0.1,0.65) -- (-0.1,0.5)  -- cycle;
\draw[line width=0.03ex*\tkzsymbls@scl] (0,0.6) -- (0.2,0.6);
\draw[line width=0.03ex*\tkzsymbls@scl] (0,0.58) -- (0.2,0.58);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Schaler#1}%
\tikzsymbolsaftersymbolinput%
}
\tikzsymbols@let{peeler}{Schaler}
```

Text hinschreiben \Reibe = \grater Text hinschreiben

```
\tkzsymbls@Declare@Robust@Command{Reibe}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Reibe#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{1.08ex*\real{\tkzsymbls@Prmtr}}{1.58ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=1ex,y=1ex, line width=0.08ex*\tkzsymbls@scl, scale=#1]%
\draw (0,0) rectangle (1,1.2);
\draw[rounded corners=0.04ex] (0.05,1.2) rectangle (0.95,1.5);
\foreach\x in {0.2,0.4,0.6,0.8}
\foreach\y in {0.2,0.4, 0.6, 0.8, 1}
\fill (\x,\y) circle (0.05ex);
\end{tikzpicture}%
```

```
\fi%
}%
\tikzsymbols@use@box{Reibe#1}%
\tikzsymbolsaftersymbolinput%
}
\tikzsymbols@let{grater}{Reibe}
```

`\Flasche = \bottle` Text hinschreiben

```
\tkzsymbls@Declare@Robust@Command{Flasche}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Flasche#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{0.78ex*\real{\tkzsymbls@Prmtr}}{1.58ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=1ex,y=1ex, line width=0.08ex, rounded corners=0.08ex*\tkzsymbls@scl, scal
\draw (0, 1.5) -- (0,1.2) -- (-0.15,0.8) -- (-0.15,0) --++ (0.6,0) --++ (0,0.8) --++ (-0.15,0.4
%\draw (-0.15,0.8) -- (0.45,0.8);
%\draw (-0.15,0.3) -- (0.45,0.3);
\draw[opacity=00] (-0.2,0) --++ (0.7,0);
\end{tikzpicture}\fi%
}%
\tikzsymbols@use@box{Flasche#1}%
\tikzsymbolsaftersymbolinput%
}
\tikzsymbols@let{bottle}{Flasche}
```

6.2 Emoticonscode

`\Sadey \dSadey` Another name of Sadey is Frowny, but I named it Sadey because there are enough Frownys in the world. All "3D" Emoticons start with `\d...`, and all Emoticons end with an "ey" (exception: "Cat", "Ninja", and else). The "default color" of the 2D Emoticons is `opacity=0`, it's useful for `\colorbox{yellow}{\Sadey}` which leads to ☹ instead of ☹ (with default=white).

```
\tkzsymbls@Declare@Robust@Commandx{Sadey}[2][1=1, 2={opacity=0}, usedefault]{%
\tikzsymbols@ifsaveboxundefined{Sadey#1#2}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.704ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.09ex*\tkzsymbls@scl,scale=#1]
\fill[#2, line width=0.1ex*\tkzsymbls@scl] (0,0) circle (0.33);
\draw[line width=0.12ex*\tkzsymbls@scl] (0,0) circle (0.33);
\fill (0.1,0.1) circle (0.05);
\fill (-0.1,0.1) circle (0.05);
\draw (-0.2,-0.15) .. controls (-0.1,-0.06) and (0.1,-0.06) .. (0.2,-0.15);
```

```
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Sadey#1#2}%
\tikzsymbolsaftersymbolinput%
}
\tkzsymbls@Declare@Robust@Commandx{dSadey}[2][1=1,2=yellow,usedefault]{%
\tikzsymbols@ifsaveboxundefined{dSadey#1#2}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.584ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.09ex*\tkzsymbls@scl,scale=#1]
\shade[ball color=#2] (0,0) circle (0.33);
\shade[ball color=black] (0.1,0.1) circle (0.05);
\shade[ball color=black] (-0.1,0.1) circle (0.05);
\draw[black] (-0.2,-0.15) .. controls (-0.1,-0.06) and (0.1,-0.06) .. (0.2,-0.15);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{dSadey#1#2}%
\tikzsymbolsaftersymbolinput%
}
```

`\Annoey \dAnnoey` An annoyed Smiley -_-

```
\tkzsymbls@Declare@Robust@Commandx{Annoey}[2][1=1,2={opacity=0},usedefault]{%
\tikzsymbols@ifsaveboxundefined{Annoey#1#2}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.704ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.09ex*\tkzsymbls@scl,scale=#1]
\fill[#2, line width=0.12ex*\tkzsymbls@scl] (0,0) circle (0.33);
\draw[line width=0.12ex*\tkzsymbls@scl] (0,0) circle (0.33);
\draw (0.08,0.1) -- (0.22,0.1);
\draw (-0.08,0.1) -- (-0.22,0.1);
\draw (-0.2,-0.1) -- (0.2,-0.1);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Annoey#1#2}%
\tikzsymbolsaftersymbolinput%
}
\tkzsymbls@Declare@Robust@Commandx{dAnnoey}[2][1=1,2=yellow,usedefault]{%
\tikzsymbols@ifsaveboxundefined{dAnnoey#1#2}{%
\set@tkzsymblsscl{#1}%
```

```
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.584ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.09ex*\tkzsymbls@scl,scale=#1]
\shade[ball color=#2] (0,0) circle (0.33);
\draw[black] (0.08,0.1) -- (0.22,0.1);
\draw[black] (-0.08,0.1) -- (-0.22,0.1);
\draw[black] (-0.2,-0.1) -- (0.2,-0.1);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{dAnnoey#1#2}%
\tikzsymbolsaftersymbolinput%
}
```

`\Smiley \dSmiley` A normal Smiley

```
\ifKV@tikzsymbols@marvosym\relax\else%
 \tkzsymbls@Declare@Robust@Commandx{Smiley}[2][1=1,2={opacity=0} ,usedefault]{%
 \tikzsymbols@ifsaveboxundefined{Smiley#1#2}{%
 \set@tkzsymblsscl{#1}%
  \iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.704ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.12ex*\tkzsymbls@scl, scale=#1]
\fill[#2] (0,0) circle (0.33);
\draw (0,0) circle (0.33);
\fill (-0.1,0.1) circle (0.05);
\fill (0.1,0.1) circle (0.05);
\draw (-0.2,-0.1) .. controls (-0.1,-0.2) and (0.1,-0.2) .. (0.2,-0.1);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Smiley#1#2}%
\tikzsymbolsaftersymbolinput%
}%
\fi
\tkzsymbls@Declare@Robust@Commandx{dSmiley}[2][1=1,2=yellow,usedefault]{%
\tikzsymbols@ifsaveboxundefined{dSmiley#1#2}{%
\set@tkzsymblsscl{#1}%
 \iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.584ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
 \begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.1ex*\tkzsymbls@scl,scale=#1]
\shade[ball color=#2] (0,0) circle (0.33);
```

```
\shade[ball color=black] (-0.1,0.1) circle (0.05);
\shade[ball color=black] (0.1,0.1) circle (0.05);
\draw[black] (-0.2,-0.1) .. controls (-0.1,-0.2) and (0.1,-0.2) .. (0.2,-0.1);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{dSmiley#1#2}%
\tikzsymbolsaftersymbolinput%
}
```

`\Laughey \dLaughey` A laughing Smiley

```
\tkzsymbls@Declare@Robust@Commandx{Laughey}[3][1=1,2={opacity=0},3={opacity=0} ,usedefault]{%
\tikzsymbols@ifsaveboxundefined{Laughey#1#2#3}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.704ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.09ex*\tkzsymbls@scl,scale=#1]
\fill[#2,line width=0.12ex*\tkzsymbls@scl] (0,0) circle (0.33);
\draw[line width=0.12ex*\tkzsymbls@scl] (0,0) circle (0.33);
\draw (-0.09,0.06) .. controls (-0.11,0.16) and (-0.17,0.16) .. +(-0.1,0);
\draw (0.09,0.06) .. controls (0.11,0.16) and (0.17,0.16) .. +(0.1,0);
\fill[#3,rounded corners=0.1ex*\tkzsymbls@scl, yshift=-0.5]
(-0.22,-0.0) .. controls (-0.13,-0.23) and (0.13,-0.23) .. (0.22,-0.0) -- cycle;
\draw[rounded corners=0.1ex*\tkzsymbls@scl, yshift=-0.5]
(-0.22,-0.0) .. controls (-0.13,-0.23) and (0.13,-0.23) .. (0.22,-0.0) -- cycle;
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Laughey#1#2#3}%
\tikzsymbolsaftersymbolinput%
}
\tkzsymbls@Declare@Robust@Commandx{dLaughey}[3][1=1,2=yellow, 3=red ,usedefault]{%
\tikzsymbols@ifsaveboxundefined{dLaughey#1#2#3}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.584ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.09ex*\tkzsymbls@scl,scale=#1]
\fill[ball color=#2,line width=0.12ex*\tkzsymbls@scl] (0,0) circle (0.33);
\draw (-0.09,0.06) .. controls (-0.11,0.16) and (-0.17,0.16) .. +(-0.1,0);
\draw (0.09,0.06) .. controls (0.11,0.16) and (0.17,0.16) .. +(0.1,0);
\shade[ball color=#3, rounded corners=0.1ex*\tkzsymbls@scl, yshift=-0.3]
 (-0.25,-0.0) .. controls (-0.13,-0.26) and (0.13,-0.26) .. (0.25,-0.0) -- cycle;
\end{tikzpicture}%
\fi%
```

```
}%
\tikzsymbols@use@box{dLaughey#1#2#3}%
\tikzsymbolsaftersymbolinput%
}
```

`\Neutrey \dNeutrey` neutral Smiley :|

```
\tkzsymbls@Declare@Robust@Commandx{Neutrey}[2][1=1, 2={opacity=0}, usedefault]{%
\tikzsymbols@ifsaveboxundefined{Neutrey#1#2}{%
 \set@tkzsymblsscl{#1}%
 \iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.704ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.09ex*\tkzsymbls@scl,scale=#1]
\fill[#2,line width=0.12ex*\tkzsymbls@scl] (0,0) circle (0.33);
\draw[line width=0.12ex*\tkzsymbls@scl] (0,0) circle (0.33);
\fill (0.1,0.1) circle (0.05);
\fill (-0.1,0.1) circle (0.05);
\draw (-0.2,-0.1) -- (0.2,-0.1);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Neutrey#1#2}%
\tikzsymbolsaftersymbolinput%
}
\tkzsymbls@Declare@Robust@Commandx{dNeutrey}[2][1=1,2=yellow,usedefault]{%
\tikzsymbols@ifsaveboxundefined{dNeutrey#1#2}{%
 \set@tkzsymblsscl{#1}%
 \iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.584ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.09ex*\tkzsymbls@scl,scale=#1]
\shade[ball color=#2] (0,0) circle (0.33);
\shade[ball color=black] (0.1,0.1) circle (0.05);
\shade[ball color=black] (-0.1,0.1) circle (0.05);
\draw[black] (-0.2,-0.1) -- (0.2,-0.1);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{dNeutrey#1#2}%
\tikzsymbolsaftersymbolinput%
}
```

`\Winkey \dWinkey` ;)

`\oldWinkey \olddWinkey`

```
\tkzsymbls@Declare@Robust@Commandx{Winkey}[2][1=1,2={opacity=0} ,usedefault]{%
\tikzsymbols@ifsaveboxundefined{Winkey#1#2}{%
\set@tkzsymblsscl{#1}%
```

```
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.704ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.12ex*\tkzsymbls@scl,scale=#1]
\fill[#2] (0,0) circle (0.33);
\draw(0,0) circle (0.33);
\draw(0.17,0.1) -- (0.05,0.1);
\fill (-0.1,0.1) circle (0.05);
\draw (-0.15,-0.15) .. controls (-0.05,-0.2) and (0.15,-0.2) .. (0.19,0);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Winkey#1#2}%
\tikzsymbolsaftersymbolinput%
}
\tkzsymbls@Declare@Robust@Commandx{oldWinkey}[2][1=1,2={opacity=0} ,usedefault]{%
\tikzsymbols@ifsaveboxundefined{oldWinkey#1#2}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.704ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.12ex*\tkzsymbls@scl,scale=#1]
\fill[#2] (0,0) circle (0.33);
\draw(0,0) circle (0.33);
\draw(0.17,0.1) -- (0.05,0.1);
\fill (-0.1,0.1) circle (0.05);
\draw (-0.2,-0.1) .. controls (-0.1,-0.2) and (0.15,-0.2) .. (0.2,0);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{oldWinkey#1#2}%
\tikzsymbolsaftersymbolinput%
}
\tkzsymbls@Declare@Robust@Commandx{dWinkey}[2][1=1,2=yellow,usedefault]{%
\tikzsymbols@ifsaveboxundefined{dWinkey#1#2}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.584ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.12ex*\tkzsymbls@scl,scale=#1]
\shade[ball color=#2] (0,0) circle (0.33);
\draw[black] (0.17,0.1) -- (0.05,0.1);
\shade[ball color=black] (-0.1,0.1) circle (0.05);
\draw[black] (-0.15,-0.15) .. controls (-0.05,-0.2) and (0.15,-0.2) .. (0.19,0);
```

```
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{dWinkey#1#2}%
\tikzsymbolsaftersymbolinput%
}
\tkzsymbls@Declare@Robust@Commandx{olddWinkey}[2][1=1,2=yellow,usedefault]{%
\tikzsymbols@ifsaveboxundefined{olddWinkey#1#2}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.584ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.12ex*\tkzsymbls@scl,scale=#1]
\shade[ball color=#2] (0,0) circle (0.33);
\draw(0.17,0.1) -- (0.05,0.1);
\shade[ball color=black] (-0.1,0.1) circle (0.05);
\draw[black] (-0.2,-0.1) .. controls (-0.1,-0.2) and (0.15,-0.2) .. (0.2,0);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{olddWinkey#1#2}%
\tikzsymbolsaftersymbolinput%
}
```

`\Sey` `\dSey` I can't think of a better name `:S`

```
\tkzsymbls@Declare@Robust@Commandx{Sey}[2][1=1,2={opacity=0} ,usedefault]{%
\tikzsymbols@ifsaveboxundefined{Sey#1#2}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.704ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.09ex*\tkzsymbls@scl,scale=#1]
\fill[#2, line width=0.12ex*\tkzsymbls@scl] (0,0) circle (0.33);
\draw[line width=0.12ex*\tkzsymbls@scl] (0,0) circle (0.33);
\fill (0.1,0.1) circle (0.05);
\fill (-0.1,0.1) circle (0.05);
\draw (-0.2,-0.08) .. controls (-0.0,-0.2) and (0.0,0) .. (0.2,-0.12);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Sey#1#2}%
\tikzsymbolsaftersymbolinput%
}
\tkzsymbls@Declare@Robust@Commandx{dSey}[2][1=1,2=yellow ,usedefault]{%
\tikzsymbols@ifsaveboxundefined{dSey#1#2}{%
\set@tkzsymblsscl{#1}%
```

```
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.584ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.09ex*\tkzsymbls@scl,scale=#1]
\shade[ball color=#2] (0,0) circle (0.33);
\shade[ball color=black] (0.1,0.1) circle (0.05);
\shade[ball color=black] (-0.1,0.1) circle (0.05);
\draw[black] (-0.2,-0.08) .. controls (-0.0,-0.2) and (0.0,0) .. (0.2,-0.12);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{dSey#1#2}%
\tikzsymbolsaftersymbolinput%
}
```

`\Xey \dXey` I can't think of a better name again.

```
\tkzsymbls@Declare@Robust@Commandx{Xey}[2][1=1, 2={opacity=0}, usedefault]{%
\tikzsymbols@ifsaveboxundefined{Xey#1#2}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.704ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.09ex*\tkzsymbls@scl,scale=#1]
\fill[#2, line width=0.12ex*\tkzsymbls@scl] (0,0) circle (0.33);
\draw[line width=0.12ex*\tkzsymbls@scl] (0,0) circle (0.33);
\draw (0.05,0.05) -- ++ (0.1,0.1);
\draw (0.15,0.05) -- ++ (-0.1,0.1);
\draw (-0.05,0.05) -- ++ (-0.1,0.1);
\draw (-0.15,0.05) -- ++ (0.1,0.1);
\draw (-0.2,-0.15) .. controls (-0.1,-0.06) and (0.1,-0.06) .. (0.2,-0.15);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Xey#1#2}%
\tikzsymbolsaftersymbolinput%
}
\tkzsymbls@Declare@Robust@Commandx{dXey}[2][1=1, 2={yellow}, usedefault]{%
\tikzsymbols@ifsaveboxundefined{dXey#1#2}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.584ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.09ex*\tkzsymbls@scl,scale=#1]
\fill[ball color=#2, line width=0.12ex*\tkzsymbls@scl] (0,0) circle (0.33);
```

```
\draw (0.05,0.05) -- ++ (0.1,0.1);
\draw (0.15,0.05) -- ++ (-0.1,0.1);
\draw (-0.05,0.05) -- ++ (-0.1,0.1);
\draw (-0.15,0.05) -- ++ (0.1,0.1);
\draw (-0.2,-0.15) .. controls (-0.1,-0.06) and (0.1,-0.06) .. (0.2,-0.15);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{dXey#1#2}%
\tikzsymbolsaftersymbolinput%
}
```

`\Innocey \dInnocey` An innocent Smiley

```
\tkzsymbls@Declare@Robust@Commandx{Innocey}[3][1=1,2={opacity=0},3=yellow ,usedefault]{%
\tikzsymbols@ifsaveboxundefined{Innocey#1#2#3}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{1.73ex*\real{\tkzsymbls@Prmtr}}{1.909ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.12ex*\tkzsymbls@scl,scale=#1]
\fill[#2] (0,0) circle (0.33);
\draw (0,0) circle (0.33);
\fill (-0.1,0.1) circle (0.05);
\fill (0.1,0.1) circle (0.05);
\draw (-0.2,-0.1) .. controls (-0.1,-0.2) and (0.1,-0.2) .. (0.2,-0.1);
\draw[#3, line width=0.095ex*\tkzsymbls@scl] (0.32,0.31) arc (0:360:0.32 and 0.1);
\draw[line width=0.05ex*\tkzsymbls@scl] (0.3,0.31) arc (0:360:0.3 and 0.07);
\draw[line width=0.05ex*\tkzsymbls@scl] (0.35,0.31) arc (0:360:0.35 and 0.12);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Innocey#1#2#3}%
\tikzsymbolsaftersymbolinput%
}
\tkzsymbls@Declare@Robust@Command{wInnocey}[1][1]{\Innocey[#1][opacity=0][white]}
\tkzsymbls@Declare@Robust@Commandx{dInnocey}[3][1=1,2=yellow,3=yellow,usedefault]{%
\tikzsymbols@ifsaveboxundefined{dInnocey#1#2#3}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{1.73ex*\real{\tkzsymbls@Prmtr}}{1.849ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.12ex*\tkzsymbls@scl,scale=#1]
\shade[ball color=#2] (0,0) circle (0.33);
\shade[ball color=black] (-0.1,0.1) circle (0.05);
\shade[ball color=black] (0.1,0.1) circle (0.05);
\draw[black] (-0.2,-0.1) .. controls (-0.1,-0.2) and (0.1,-0.2) .. (0.2,-0.1);
```

```
\draw[color=#3!97!black, line width=0.1ex*\tkzsymbls@scl]
   (0.32,0.31) arc (0:360:0.32 and 0.1);
\draw[line width=0.05ex*\tkzsymbls@scl] (0.3,0.31) arc (0:360:0.3 and 0.07);
\draw[line width=0.05ex*\tkzsymbls@scl] (0.35,0.31) arc (0:360:0.35 and 0.12);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{dInnocey#1#2#3}%
\tikzsymbolsaftersymbolinput%
}
```

`\Cooley \dCooley` Don't know what I shall write here.

```
\tkzsymbls@Declare@Robust@Commandx{Cooley}[2][1=1,2={opacity=0} ,usedefault]{%
\tikzsymbols@ifsaveboxundefined{Cooley#1#2}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.704ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.12ex*\tkzsymbls@scl,scale=#1]
\fill[#2] (0,0) circle (0.33);
\draw (0,0) circle (0.33);
\fill[rounded corners=0.1ex*\tkzsymbls@scl]
   (0.24,0.15) -- (0.01,0.15) -- (0.01,0) -- (0.24,0) -- cycle;
\fill[rounded corners=0.1ex*\tkzsymbls@scl]
   (-0.24,0.15) -- (-0.01,0.15) -- (-0.01,0) -- (-0.24,0) -- cycle;
\draw (-0.2,-0.1) .. controls (-0.1,-0.2) and (0.1,-0.2) .. (0.2,-0.1);
\draw (0.02,0.1) -- (-0.02,0.1);
\draw (-0.2,0.1) -- (-0.3,0.13);
\draw (0.2,0.1) -- (0.3,0.13);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Cooley#1#2}%
\tikzsymbolsaftersymbolinput%
}
\tkzsymbls@Declare@Robust@Commandx{dCooley}[2][1=1,2=yellow,usedefault]{%
\tikzsymbols@ifsaveboxundefined{dCooley#1#2}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.584ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.12ex*\tkzsymbls@scl,scale=#1]
\shade[ball color=#2] (0,0) circle (0.33);
\draw[black] (0.02,0.1) -- (-0.02,0.1);
\draw[black] (-0.2,0.1) -- (-0.295,0.146);
\draw[black] (0.2,0.1) -- (0.295,0.146);
```

```
\shade[ball color=black,rounded corners=0.1ex*\tkzsymbls@scl]
   (0.24,0.15) -- (0.01,0.15) -- (0.01,0) -- (0.24,0) -- cycle;
\shade[ball color=black,rounded corners=0.1ex*\tkzsymbls@scl]
   (-0.24,0.15) -- (-0.01,0.15) -- (-0.01,0) -- (-0.24,0) -- cycle;
\draw[black] (-0.2,-0.1) .. controls (-0.1,-0.2) and (0.1,-0.2) .. (0.2,-0.1);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{dCooley#1#2}%
\tikzsymbolsaftersymbolinput%
}
```

\Tongey \dTongey :P

```
\tkzsymbls@Declare@Robust@Commandx{Tongey}[3][1=1,2={opacity=0},3={opacity=0} ,usedefault]{%
\tikzsymbols@ifsaveboxundefined{Tongey#1#2#3}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.704ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.12ex*\tkzsymbls@scl,scale=#1]
\fill[#2] (0,0) circle (0.33);
\draw (0,0) circle (0.33);
\fill (-0.1,0.1) circle (0.05);
\fill (0.1,0.1) circle (0.05);
\fill[#3,line width=0.058ex*\tkzsymbls@scl, rounded corners=0.12ex*\tkzsymbls@scl]
   (0,-0.09) -- (0.05,-0.2) -- (0.16,-0.23) -- (0.2,-0.15) -- (0.19,-0.03);
\draw[line width=0.07ex*\tkzsymbls@scl, yshift=0.21ex]
   (-0.2,-0.1) .. controls (-0.1,-0.2) and (0.1,-0.2) .. (0.2,-0.1);
\draw[line width=0.058ex*\tkzsymbls@scl, rounded corners=0.12ex*\tkzsymbls@scl]
   (0,-0.09) -- (0.05,-0.2) -- (0.16,-0.23) -- (0.2,-0.15) -- (0.19,-0.03);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Tongey#1#2#3}%
\tikzsymbolsaftersymbolinput%
}
\tkzsymbls@Declare@Robust@Commandx{dTongey}[3][1=1,2=yellow,3=red,usedefault]{%
\tikzsymbols@ifsaveboxundefined{dTongey#1#2#3}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftQbox{1.584ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.12ex*\tkzsymbls@scl,scale=#1]
\shade[ball color=#2] (0,0) circle (0.33);
\shade[ball color=black] (-0.1,0.1) circle (0.05);
\shade[ball color=black] (0.1,0.1) circle (0.05);
```

```
\shade[ball color=#3,line width=0.058ex*\tkzsymbls@scl, rounded corners=0.12ex*\tkzsymbls@scl]
   (0,-0.09) -- (0.05,-0.2) -- (0.16,-0.23) -- (0.2,-0.15) -- (0.19,-0.03);
\draw[black, line width=0.058ex*\tkzsymbls@scl, rounded corners=0.12ex*\tkzsymbls@scl]
   (0,-0.09) -- (0.05,-0.2) -- (0.16,-0.23) -- (0.2,-0.15) -- (0.19,-0.03);
\draw[black, line width=0.07ex*\tkzsymbls@scl, yshift=0.21ex]
   (-0.2,-0.1) .. controls (-0.1,-0.2) and (0.1,-0.2) .. (0.2,-0.1);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{dTongey#1#2#3}%
\tikzsymbolsaftersymbolinput%
}
```

`\Nursey \dNursey` a Nurse.

```
\tkzsymbls@Declare@Robust@Commandx{Nursey}
[4][1=1,2={opacity=0},3={opacity=0},4=black,usedefault]{%
\tikzsymbols@ifsaveboxundefined{Nursey#1#2#3#4}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{1.5ex*\real{\tkzsymbls@Prmtr}}{2.19ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.3ex, y=2.3ex, line width=0.12ex*\tkzsymbls@scl,scale=#1]
\fill[#3,rounded corners=.023ex*\tkzsymbls@scl]
   (-0.3,0) -- (-0.3,0.3) -- (0,0.6) -- (0.3,0.3) -- (0.3,0);
\fill[#2] (0,0) circle (0.3);
\draw (0,0) circle (0.3);
\fill (-0.1,0.1) circle (0.05);
\fill (0.1,0.1) circle (0.05);
\draw[line width=0.09ex*\tkzsymbls@scl, yshift=0.07ex]
   (-0.2,-0.1) .. controls (-0.1,-0.2) and (0.1,-0.2) .. (0.2,-0.1);
\draw[rounded corners=.023ex*\tkzsymbls@scl]
   (-0.3,0) -- (-0.3,0.3) -- (0,0.6) -- (0.3,0.3) -- (0.3,0);
\draw[#4,line width=.046ex*\tkzsymbls@scl] (0,0.35) -- (0,0.5);
\draw[#4,line width=.046ex*\tkzsymbls@scl] (-0.05,0.45) -- (0.05,0.45);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Nursey#1#2#3#4}%
\tikzsymbolsaftersymbolinput%
}
\tkzsymbls@Declare@Robust@Commandx{dNursey}[4][1=1,2=yellow,3=white,4=red,usedefault]{%
\tikzsymbols@ifsaveboxundefined{dNursey#1#2#3#4}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{1.38ex*\real{\tkzsymbls@Prmtr}}{1.98ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
```

```
\begin{tikzpicture}[x=2.3ex, y=2.3ex, line width=0.12ex*\tkzsymbls@scl,scale=#1]
\shade[ball color=#2] (0,0) circle (0.3);
\shade[ball color=black] (-0.1,0.1) circle (0.05);
\shade[ball color=black] (0.1,0.1) circle (0.05);
\draw[black, line width=0.09ex*\tkzsymbls@scl, yshift=0.07ex]
   (-0.2,-0.1) .. controls (-0.1,-0.2) and (0.1,-0.2) .. (0.2,-0.1);
\shade[ball color=#3, rounded corners=.023ex*\tkzsymbls@scl,yshift=-0.09ex]
   (-0.3,0) -- (-0.3,0.3) -- (0,0.6) -- (0.3,0.3) -- (0.3,0) arc (0:180:0.3);
\shade[ball color=#4,line width=.046ex*\tkzsymbls@scl]
   (-0.01,0.31) -- (-0.01,0.46) -- (0.01,0.46) -- (0.01,0.31)--cycle;
\shade[ball color=#4,line width=.046ex*\tkzsymbls@scl]
   (-0.05,0.4) -- (0.05,0.4) -- (0.05,0.42)--(-0.05,0.42) -- cycle;
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{dNursey#1#2#3#4}%
\tikzsymbolsaftersymbolinput%
}
```

`\Vomey \dVomey` *Bläärgh*

```
\tkzsymbls@Declare@Robust@Commandx{Vomey}[3][1=1,2={opacity=0},3={opacity=0},usedefault]{%
\tikzsymbols@ifsaveboxundefined{Vomey#1#2#3}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{3.0335ex*\real{\tkzsymbls@Prmtr}}{1.743ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=0.58ex,y=0.58ex, line width=0.09ex*\tkzsymbls@scl,scale=#1]
\fill[#2,rounded corners=0.05ex*\tkzsymbls@scl]  (0,0)  arc (15:330:1) -- (-0.6,-0.3) -- cycle;
\draw[rounded corners=0.05ex*\tkzsymbls@scl]  (0,0)  arc (15:330:1) -- (-0.6,-0.3) -- cycle;
\draw[line width=0.05ex*\tkzsymbls@scl] (-0.5,0.3) -- (-0.3,0.1);
\fill (-0.45,0.27) arc (100:350:0.1);
\fill[#3] (1.8,-0.5)  .. controls (2.5,-0.3) and (2.8,-0.7) .. (2.5,-1) ..
   controls (3,-1) and (3,-1.7) .. (2,-1.5) .. controls (1.7,-2) and (1,-2) .. (1,-1.5) ..
   controls (0.5,-1.9) and (0.3,-1) .. (0.7,-0.9);
\fill[#3] (0,-0.4) .. controls (1,0) and (2,-1) .. (2,-1) ..
   controls (1.7,-1.2) and (1.3,-1.2) .. (1,-1) ..
   controls (0.8,-0.7) and (0.5,-0.5) .. (0,-0.4);
\draw (0,-0.4) .. controls (1,0) and (2,-1) .. (2,-1);
\draw (0,-0.4) .. controls (0.5,-0.5) and (0.8,-0.7) .. (1,-1);
\draw (1.8,-0.5)  .. controls (2.5,-0.3) and (2.8,-0.7) .. (2.5,-1) ..
   controls (3,-1) and (3,-1.7) .. (2,-1.5) .. controls (1.7,-2)
   and (1,-2) .. (1,-1.5) .. controls (0.5,-1.9) and (0.3,-1) .. (0.7,-0.9);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Vomey#1#2#3}%
\tikzsymbolsaftersymbolinput%
}
```

```
\tkzsymbls@Declare@Robust@Commandx{dVomey}[3][1=1,2=yellow,3={brown!10!olive},usedefault]{%
\tikzsymbols@ifsaveboxundefined{dVomey#1#2#3}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{3.2435ex*\real{\tkzsymbls@Prmtr}}{1.653ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=0.58ex,y=0.58ex, line width=0.09ex*\tkzsymbls@scl,scale=#1]
\shade[ball color=#2!90!brown,rounded corners=0.03ex*\tkzsymbls@scl]
   (0,0)  arc (15:330:1) -- (-0.6,-0.3) -- cycle;
\draw[black, line width=0.05ex*\tkzsymbls@scl] (-0.5,0.3) -- (-0.3,0.1);
\shade[ball color=black] (-0.45,0.27) arc (100:350:0.1);
\shade[ball color=#3] (1.8,-0.5)  .. controls (2.5,-0.3) and (2.8,-0.7) .. (2.5,-1) ..
 controls (3,-1) and (3,-1.7) .. (2,-1.5) .. controls (1.7,-2) and (1,-2) .. (1,-1.5) ..
 controls (0.5,-1.9) and (0.3,-1) .. (0.7,-0.9);
\shade[ball color=#3] (0,-0.4) .. controls (1,0) and (2,-1) .. (2,-1) .. controls
   (1.7,-1.2) and (1.3,-1.2) .. (1,-1) .. controls (0.8,-0.7) and (0.5,-0.5) .. (0,-0.4);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{dVomey#1#2#3}%
\tikzsymbolsaftersymbolinput%
}
```

`\Walley \dWalley` Well ... this Emoticon should be the visualization of the german saying "Gegen eine Wand rennen", which means something like: Not being able to solve a problem.

```
\tkzsymbls@Declare@Robust@Commandx{Walley}[3][1=1, 2={opacity=0},3={opacity=0}, usedefault]{%
\tikzsymbols@ifsaveboxundefined{Walley#1#2#3}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{2.341ex*\real{\tkzsymbls@Prmtr}}{1.674ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.09ex*\tkzsymbls@scl,scale=#1,
decoration={random steps,segment length=0.15ex*\tkzsymbls@scl, amplitude=0.1ex*\tkzsymbls@scl}]
\fill[#2, line width=0.08ex*\tkzsymbls@scl] (0,0) circle (0.28);
\draw[line width=0.08ex*\tkzsymbls@scl] (0,0) circle (0.28);
\fill[#3] (0.28,-0.33) rectangle (0.66,0.33);
\draw (0.28,-0.33) rectangle (0.66,0.33);
\draw[line width=0.06ex*\tkzsymbls@scl]
(0.28,0) --++(0.05,0.07) --++(0.03,0.02) --+
+(0.03,-0.02) --++(0.03,0.1) --++(0.03,0.02) -- (0.5,0.25);
\draw[line width=0.06ex*\tkzsymbls@scl]
(0.28,0) --++(0.06,-0.02) --++(0.04,0.06) --+
+(0.0,-0.08) --++(0.08,0.06) --++(0.03,-0.02) --+(0.08,0.02) -- (0.6,0.0);
\draw[line width=0.06ex*\tkzsymbls@scl]
(0.28,0) --++(0.03,-0.02)  --++(0.03,-0.07)  --+
+(0.03,-0.01)  --++(0.01,-0.07)  --++(0.06,0.01)  --++(0.03,-0.08)   --
```

```
(0.5,0.-0.25);
\draw[rotate=-20] (0.12,0.1) -- (0.2,0.05);
\draw[rotate=-20] (0.27,-0.1) .. controls (0.2,-0.072) and (0.1,-0.06) .. (0.,-0.1);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Walley#1#2#3}%
\tikzsymbolsaftersymbolinput%
}
\tkzsymbls@Declare@Robust@Commandx{rWalley}[3][1=1, 2={opacity=0},3={opacity=0}, usedefault]{%
\tikzsymbols@ifsaveboxundefined{rWalley#1#2#3}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{2.341ex*\real{\tkzsymbls@Prmtr}}{1.674ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.09ex*\tkzsymbls@scl,scale=#1,
decoration={random steps,segment length=0.15ex*\tkzsymbls@scl, amplitude=0.1ex*\tkzsymbls@scl}]
\fill[#2, line width=0.08ex*\tkzsymbls@scl] (0,0) circle (0.28);
\draw[line width=0.08ex*\tkzsymbls@scl] (0,0) circle (0.28);
\fill[#3] (0.28,-0.33) rectangle (0.66,0.33);
\draw (0.28,-0.33) rectangle (0.66,0.33);
\draw[decorate, line width=0.06ex*\tkzsymbls@scl] (0.28,0) -- (0.5,0.25);
\draw[decorate,line width=0.06ex*\tkzsymbls@scl] (0.28,0) -- (0.6,0.0);
\draw[decorate,line width=0.06ex*\tkzsymbls@scl] (0.28,0) -- (0.5,-0.25);
\draw[rotate=-20] (0.12,0.1) -- (0.2,0.05);
\draw[rotate=-20] (0.27,-0.1) .. controls (0.2,-0.072) and (0.1,-0.06) .. (0.,-0.1);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{rWalley#1#2#3}%
\tikzsymbolsaftersymbolinput%
}
\tkzsymbls@Declare@Robust@Commandx{dWalley}[2][1=1, 2=yellow, usedefault]{%
\tikzsymbols@ifsaveboxundefined{dWalley#1#2}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{2.4288ex*\real{\tkzsymbls@Prmtr}}{1.6008ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.09ex*\tkzsymbls@scl,scale=#1]
\shade[ball color=orange!80!black] (0.298,-0.33) rectangle (0.692,0.337);
\draw[line width=0.06ex*\tkzsymbls@scl]
(0.28,0) --++(0.05,0.07) --++(0.03,0.02) --+
+(0.03,-0.02) --++(0.03,0.1) --++(0.03,0.02) -- (0.5,0.25);
\draw[line width=0.06ex*\tkzsymbls@scl]
(0.28,0) --++(0.06,-0.02) --++(0.04,0.06) --+
+(0.0,-0.08) --++(0.08,0.06) --++(0.03,-0.02) --+(0.08,0.02) -- (0.6,0.0);
```

```
\draw[line width=0.06ex*\tkzsymbls@scl]
(0.28,0) --++(0.03,-0.02)  --++(0.03,-0.07)  --+
+(0.03,-0.01)  --++(0.01,-0.07)  --++(0.06,0.01)  --++(0.03,-0.08)   -- (0.5,0.-0.25);
\shade[ball color=#2] (-0.01,0) circle (0.31);
\draw[rotate=-20] (0.12,0.1) -- (0.2,0.05);
\draw[rotate=-20] (0.283,-0.1) .. controls (0.2,-0.072) and (0.1,-0.06) .. (0,-0.1);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{dWalley#1#2}%
\tikzsymbolsaftersymbolinput%
}
\tkzsymbls@Declare@Robust@Commandx{drWalley}[2][1=1, 2=yellow, usedefault]{%
\tikzsymbols@ifsaveboxundefined{drWalley#1#2}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{2.4288ex*\real{\tkzsymbls@Prmtr}}{1.6008ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.09ex*\tkzsymbls@scl,scale=#1,
decoration={random steps,segment length=0.15ex*\tkzsymbls@scl, amplitude=0.1ex*\tkzsymbls@scl}]
\shade[ball color=orange!80!black] (0.298,-0.33) rectangle (0.692,0.337);
\draw[decorate, line width=0.06ex*\tkzsymbls@scl] (0.298,0) -- (0.5,0.25);
\draw[decorate,line width=0.06ex*\tkzsymbls@scl] (0.298,0) -- (0.6,0.0);
\draw[decorate,line width=0.06ex*\tkzsymbls@scl] (0.298,0) -- (0.5,-0.25);
\shade[ball color=#2, line width=0.08ex*\tkzsymbls@scl] (-0.01,0) circle (0.31);
\draw[rotate=-20] (0.12,0.1) -- (0.2,0.05);
\draw[rotate=-20] (0.283,-0.1) .. controls (0.2,-0.072) and (0.1,-0.06) .. (0.,-0.1);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{drWalley#1#2}%
\tikzsymbolsaftersymbolinput%
}
```

\Cat *Miau*

```
\tkzsymbls@Declare@Robust@Commandx{Cat}[1][1=1,usedefault]{%
\tikzsymbols@ifsaveboxundefined{Cat#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{1.899ex*\real{\tkzsymbls@Prmtr}}{1.957ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=2.33ex,y=2.33ex, line width=0.093ex*\tkzsymbls@scl,scale=#1]
\draw (0,0) circle (0.3);
\draw[rounded corners=0.163ex*\tkzsymbls@scl] (-0.3,0) -- (-0.35,0.5) -- (0,0.3);
\draw[rounded corners=0.163ex*\tkzsymbls@scl] (0,0.3) -- (0.35,0.5) -- (0.3,0);
\fill (-0.15,.15) circle (0.05);
```

```
\fill (0.15,.15) circle (0.05);
\draw[rounded corners=0.175ex*\tkzsymbls@scl,yshift=-0.12ex]
   (0,0) -- (0,-0.1) -- (-0.1,-0.095);
\draw[rounded corners=0.175ex*\tkzsymbls@scl,yshift=-0.12ex]
   (0,0) -- (0,-0.1) -- (0.1,-0.095);
\draw[rounded corners=.12ex*\tkzsymbls@scl,yshift=-.15ex,
         line width=0.03em*\real{0.9}*\tkzsymbls@scl]
(-0.1,0.1) -- (0,0) -- (0.1,0.1) -- cycle ;
\draw[line width=0.035ex*\tkzsymbls@scl]
   (-0.1,-0.05)..controls(-0.25,0)and(-0.35,0).. (-0.4,-0.05);
\draw[line width=0.035ex*\tkzsymbls@scl](-0.1,-0.05)..
   controls(-0.25,-0.01)and(-0.35,-0.09).. (-0.4,-0.14);
\draw[line width=0.035ex*\tkzsymbls@scl](-0.1,-0.05)..
   controls(-0.25,-0.045)and(-0.35,-0.13).. (-0.4,-0.23);
\draw[line width=0.035ex*\tkzsymbls@scl]
   (0.1,-0.05)..controls(0.25,0)and(0.35,0).. (0.4,-0.05);
\draw[line width=0.035ex*\tkzsymbls@scl]
   (0.1,-0.05)..controls(0.25,-0.01)and(0.35,-0.09).. (0.4,-0.14);
\draw[line width=0.035ex*\tkzsymbls@scl]
   (0.1,-0.05)..controls(0.25,-0.045)and(0.35,-0.13).. (0.4,-0.23);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Cat#1}%
\tikzsymbolsaftersymbolinput%
}
```

`\Ninja` `\dNinja` A Ninja.

```
\tkzsymbls@Declare@Robust@Commandx{Ninja}[4][1=1, 2=black, 3=red, 4=white, usedefault]{%
\tikzsymbols@ifsaveboxundefined{Ninja#1#2#3#4}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{2.149ex*\real{\tkzsymbls@Prmtr}}{1.717ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\def\Black@is@Black{black}%
\def\Black@or@not@Black{#2}%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.09ex*\tkzsymbls@scl,scale=#1,
decoration={random steps,segment length=0.1ex*\tkzsymbls@scl, amplitude=0.1ex*\tkzsymbls@scl}]
\fill[#2] (0,0) circle (0.33);
%\draw (-0.2,-0.125) -- ++(0.4,0);
\fill[decoration={random steps,segment length=0.1ex*\tkzsymbls@scl,
  amplitude=0.01ex*\tkzsymbls@scl}, decorate,#3]
(-0.33,0) -- (0.33,0)  -- (0.23,0.23) -- (-0.23,0.23) -- cycle;
\ifx\Black@or@not@Black\Black@is@Black
   \draw[line width=0.08ex*\tkzsymbls@scl] (0,0) circle (0.33);\fi
\fill[#3] (0,0.1) -- (-0.33,0) -- (-0.26,0.23);
\fill[#3] (0.3465,0) arc (0:42:0.34 and 0.345)  --
  (0.2,0.23)-- (0.31,0.0) -- cycle;
```

```
\fill[#3] (-0.3465,0) arc (0:-42:-0.34 and -0.345)  --
   (-0.2,0.23)-- (-0.31,0.0) -- cycle;
\fill[#4] (0.129,0.1425) arc (55:-180:.05);
\fill[#4] (-0.129,0.1425) arc (-55:180:-.05);
\draw[decorate,decoration={snake,amplitude=.1ex*\tkzsymbls@scl,
 segment length=0.55ex*\tkzsymbls@scl}, #3]
  (0.26,0.21) -- (0.5,0.35);
\draw[decorate,decoration={snake,amplitude=.1ex*\tkzsymbls@scl,
 segment length=0.55ex*\tkzsymbls@scl}, #3]
  (0.26,0.21) -- (0.53,0.1);
\ifx\Black@or@not@Black\Black@is@Black
\else\draw[line width=0.08ex*\tkzsymbls@scl] (0,0) circle (0.33);\fi
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Ninja#1#2#3#4}%
\tikzsymbolsaftersymbolinput%
}
\tkzsymbls@Declare@Robust@Commandx{dNinja}[4][1=1, 2=black, 3=red, 4=white, usedefault]{%
\tikzsymbols@ifsaveboxundefined{dNinja#1#2#3#4}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{2.1498ex*\real{\tkzsymbls@Prmtr}}{1.7178ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\def\Black@is@Black{black}%
\def\Black@or@not@Black{#2}%
\begin{tikzpicture}[x=2.4ex, y=2.4ex, line width=0.09ex*\tkzsymbls@scl,scale=#1,
decoration={random steps,segment length=0.1ex*\tkzsymbls@scl, amplitude=0.1ex*\tkzsymbls@scl}]
\draw[ decorate,decoration={snake,amplitude=.1ex*\tkzsymbls@scl,
segment length=0.55ex*\tkzsymbls@scl},decorate, #3!50!black]
   (0.26,0.21) -- (0.5,0.35);
\draw[ decorate,decoration={snake,amplitude=.1ex*\tkzsymbls@scl,
segment length=0.5ex*\tkzsymbls@scl},decorate, #3!50!black]
   (0.26,0.21) -- (0.53,0.1);
\shade[ball color=#2] (0,0) circle (0.347);
%\draw (-0.2,-0.125) -- ++(0.4,0);
\fill[decoration={random steps,segment length=0.1ex*\tkzsymbls@scl,
 amplitude=0.01ex*\tkzsymbls@scl},ball color=#3]
decorate  {(-0.33,0) -- (0.3465,0)  }
        {arc (0:42:0.34 and 0.345)}
decorate  {-- (-0.25,0.24)}
     { arc (-42:0:-0.375 and -0.345)};
\shade[ball color=#4] (0.129,0.1425) arc (55:-180:.05);
\shade[ball color=#4] (-0.129,0.1425) arc (-55:180:-.05);
\shade[top color=#4!80!black, bottom color=#4] (0.129,0.1425) arc (55:-180:.05);
\shade[top color=#4!80!black, bottom color=#4] (-0.129,0.1425) arc (-55:180:-.05);
\end{tikzpicture}%
\fi%
```

```
}%
\tikzsymbols@use@box{dNinja#1#2#3#4}%
\tikzsymbolsaftersymbolinput%
}
```

\NiceReapey Not very well made. But it's better than nothing

```
\tkzsymbls@Declare@Robust@Commandx{NiceReapey}[2][1=1,2={black!20!white},usedefault]{%
\tikzsymbols@ifsaveboxundefined{NiceReapey#1#2}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{(1.1067em+0.07ex)*\real{\tkzsymbls@Prmtr}}
                     {(0.693em+0.07ex)*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=0.11em,y=0.11em, line width=0.07ex*\tkzsymbls@scl,scale=#1]
\draw[] (1.7,-1) arc (360:180:1.7 and 2)
   arc (260:110:1.5 and 2) .. controls (-1,3.3) and (1,3.3) .. (1.9,2.97)
   arc (260:100:-1.3 and -2) -- cycle;
\fill[#2] (3,3) .. controls (5,3) and (6,2) .. (7,1.5) -- (3,1.5) -- cycle;
\draw (3,-3) -- (3,3) .. controls (5,3) and (6,2) .. (7,1.5) -- (3,1.5);
\draw (0,-1.5) circle (1 and 0.5);
\draw[line width=0.04ex*\tkzsymbls@scl] (-0.2,-1) -- (-0.2,-2);
\draw[line width=0.04ex*\tkzsymbls@scl] (0.2,-1) -- (0.2,-2);
\draw[line width=0.04ex*\tkzsymbls@scl] (0.6,-1) -- (0.6,-2);
\draw[line width=0.04ex*\tkzsymbls@scl] (-0.6,-1) -- (-0.6,-2);
\draw[line width=0.04ex*\tkzsymbls@scl] (-1,-1.5) -- (1,-1.5);
\fill (1.25,1.25) circle ( 0.5 and 0.75);
\fill (-1.25,1.25) circle ( 0.5 and 0.75);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{NiceReapey#1#2}
\tikzsymbolsaftersymbolinput%
}
```

6.3 Other symbols(s)

\tikzsymbols@Strichmaxerl@XCheck This macro is needed for \Strichmaxerl. It's not easy to explain why it does what it does, but . . . uhm . . . it has something to do with mathematics and why the plain vanilla rectangle has always the correct proportions.

It is important for having \Strichmaxerl's plain vanilla rectangle the correct size. Well, at first this macro checks if #1 is greater than 0. If it is, it checks if #1 is smaller than 0.18 (radius of the Strichmaxerl's head). If it is, it sets #1 to 0 (it is smaller than the head-radius and is therefore not needed). If it is not, it subtracts 0.18 from #1 (we only want the length which overhangs the head). Else . . .

```
\newcommand{\tikzsymbols@Strichmaxerl@XCheck}[1]{%
\ifdim #1 pt > 0pt%
   \ifdim #1 pt < 0.18pt%
```

```
      \pgfmathsetmacro{#1}{0}%
    \else%
      \pgfmathsetmacro{#1}{#1-0.18}%
   \fi%
 \else%
   \ifdim #1 pt >-0.18pt%
      \pgfmathsetmacro{#1}{0}%
    \else%
      \pgfmathsetmacro{#1}{#1+0.18}%
   \fi%
\fi%
}
```

`ools@Strichmaxerl@ifSmallerZero` Checks if something is smaller than zero (< 0).

```
\newcommand{\tikzsymbols@Strichmaxerl@ifSmallerZero}[1]{%
\ifdim #1 pt < 0pt%
      \pgfmathsetmacro{#1}{0}%
\fi%
}
```

`\Strichmaxerl` My first symbol: a Strichmaxerl. And one of the most complex symbols in this package.

```
\tkzsymbls@Declare@Robust@Commandx{Strichmaxerl}[5][1=1,2=-22,3=22,4=27,5=-27,usedefault]{%
\tikzsymbols@ifsaveboxundefined{Strichmaxerl#1#2#3#4#5}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
```

Now we have to calculate the length and the height of the separate parts of the `\Strichmaxerl`. At first the lengths (they have all an `x` in the name).
`LA` for "linker Arm" (*left arm*).
`RA` for "rechter Arm" (*right arm*).
`LB` for "linkes Bein" (*left leg*).
`RB` for "rechtes Bein" (*right leg*).

```
\pgfmathsetmacro{\tikzsymbols@x@LA}{-0.27*cos(#2)}%
\pgfmathsetmacro{\tikzsymbols@x@RA}{0.27*cos(#3)}%
\pgfmathsetmacro{\tikzsymbols@x@LB}{0.34*sin(#4)}%
\pgfmathsetmacro{\tikzsymbols@x@RB}{0.34*sin(#5)}%
```

Now the height (`y`):
`LA` for "linker Arm" (*left arm*).
`RA` for "rechter Arm" (*right arm*).
`LB` for "linkes Bein" (*left leg*).
`RB` for "rechtes Bein" (*right leg*).

```
\pgfmathsetmacro{\tikzsymbols@y@LA}{0.27*sin(#2)}%
\pgfmathsetmacro{\tikzsymbols@y@RA}{-0.27*sin(#3)}%
\pgfmathsetmacro{\tikzsymbols@y@LB}{0.34*cos(#4)}%
\pgfmathsetmacro{\tikzsymbols@y@RB}{0.34*cos(#5)}%
```

Well then, lets start our calculations. Firstly the length.

We use the `\tikzsymbols@Strichmaxerl@XCheck` to check if ... (see above).

```
\tikzsymbols@Strichmaxerl@XCheck{\tikzsymbols@x@LA}%
\tikzsymbols@Strichmaxerl@XCheck{\tikzsymbols@x@RA}%
\tikzsymbols@Strichmaxerl@XCheck{\tikzsymbols@x@LB}%
\tikzsymbols@Strichmaxerl@XCheck{\tikzsymbols@x@RB}%
```

We want the greatest and the smallest length for our rectangle. To evaluate them, we define `\tikzsymbols@x@max` and `\tikzsymbols@x@min` using tikz `max()` and `min()`. The 0 is very important: if for example all lengths are negative, the greatest number is 0. (Sorry, I don't want to explain it, it has something to do with math).

```
\pgfmathsetmacro{\tikzsymbols@x@max}
    {max(0,\tikzsymbols@x@LA,\tikzsymbols@x@RA,\tikzsymbols@x@LB,\tikzsymbols@x@RB)}%
\pgfmathsetmacro{\tikzsymbols@x@min}
    {min(0,\tikzsymbols@x@LA,\tikzsymbols@x@RA,\tikzsymbols@x@LB,\tikzsymbols@x@RB)}%
```

Finished the length. Now we calculate our height. Arms and legs more or less separate.

Arms: First we subtract 0.2 (= adding −0.2) (torso length)

```
 \pgfmathsetmacro{\tikzsymbols@y@LA}{\tikzsymbols@y@LA-0.2}%
 \pgfmathsetmacro{\tikzsymbols@y@RA}{\tikzsymbols@y@RA-0.2}%
```

Arms and Legs: if they are smaller than 0, make them 0.

```
\tikzsymbols@Strichmaxerl@ifSmallerZero{\tikzsymbols@y@LA}%
\tikzsymbols@Strichmaxerl@ifSmallerZero{\tikzsymbols@y@RA}%
\tikzsymbols@Strichmaxerl@ifSmallerZero{\tikzsymbols@y@LB}%
\tikzsymbols@Strichmaxerl@ifSmallerZero{\tikzsymbols@y@RB}%
```

And find the greatest number.

```
\pgfmathsetmacro{\tikzsymbols@y@max}
{max(\tikzsymbols@y@LA,\tikzsymbols@y@RA,\tikzsymbols@y@LB,\tikzsymbols@y@RB)}%
```

For the box-length we calculate:

$$(0.606ex+1.35ex(\text{x=1.35ex})*(\text{greatest (positive) length})-1.35*(\text{smallest (negative) length}))*\text{scale} \tag{1}$$

For the height:

$$(1.173ex + 1.35ex * \text{greatest height}) * \text{scale} \tag{2}$$

```
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{(0.606ex+1.35ex*\real{\tikzsymbols@x@max}
   -1.35ex*\real{\tikzsymbols@x@min})*\real{\tkzsymbls@Prmtr}}%
{(1.173ex+1.35ex*\real{\tikzsymbols@y@max})*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
 \begin{tikzpicture}[line width=0.12ex*\tkzsymbls@scl, scale=#1, x=1.35ex, y=1.35ex]
 \draw[rotate around={#5:(0.15,0.2)}]  (0.15,0.2) -- (0.15,-0.14);
 \draw[rotate around={#4:(0.15,0.2)}]  (0.15,0.2) -- (0.15,-0.14);
 \draw (.15,.2) -- (.15,.4);
 \draw[rotate around={#3:(.15,.4)}] (0.15,0.4) -- (0.42,0.4);
```

```
 \draw[rotate around={#2:(.15,.4)}] (0.15,0.4) -- (-0.12,0.4);
 \draw (.15, .4) -- (.15, .53);
 \draw (.15,.8) circle (0.18);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Strichmaxerl#1#2#3#4#5}%
\tikzsymbolsaftersymbolinput%
}
\tkzsymbls@Declare@Robust@Command{Person}{%
\PackageWarning{tikzsymbols}{Command '\protect\Person' is obsolete,
\MessageBreak Please use '\protect\Strichmaxerl' instead.\MessageBreak}%
\Strichmaxerl%
}
```

`\Candle` A burning candle

```
\tkzsymbls@Declare@Robust@Command{Candle}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Candle#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{0.64ex*\real{\tkzsymbls@Prmtr}}{(1.255ex+2.2pt)*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=1ex, y=1ex, scale=#1, line width=0.07ex*\tkzsymbls@scl]
\draw[rounded corners=0.04ex*\tkzsymbls@scl] (0,0) -- (0.2,0) -- +(0,1) -- (0,1) -- cycle;
\draw[line width=0.05ex*\tkzsymbls@scl] (0.1,1) -- (0.1,1.2);
\draw[xshift=0.95, yshift=2.2, line width=0.04ex*\tkzsymbls@scl]
   (-0.1,0.6) .. controls (-0.4,0.8) and (-0.1,1) ..  (-0.1,1.2);
\draw [xshift=0.95, yshift=2.2, line width=0.04ex*\tkzsymbls@scl]
   (-0.1,0.6) .. controls (0.2,0.8) and (-0.1,1) ..  (-0.1,1.2);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Candle#1}%
\tikzsymbolsaftersymbolinput%
}
```

`\Fire` Just a fire.

```
\tkzsymbls@Declare@Robust@Command{Fire}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Fire#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{1.576ex*\real{\tkzsymbls@Prmtr}}{1.639ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=1ex,y=1ex,  line width=0.07ex*\tkzsymbls@scl,rotate=45, scale=#1]
\fill (-0.05,0) -- (0.05,0) -- (0.05,0.95) -- (-0.05,0.95) -- cycle;
\fill (-0.74,0.7) -- (0.19,0.7) -- (0.19,0.8) -- (-0.74,0.8) -- cycle;
```

```
\fill[rotate=-20, xshift=-1.3, yshift=-0.1]
  (-0.05,0.07) -- (0.05,0.07) -- (0.05,0.9) -- (-0.05,0.9) -- cycle;
\fill[rotate=-70, xshift=-3.3, yshift=-2.3]
  (-0.05,0.07) -- (0.05,0.07) -- (0.05,0.9) -- (-0.05,0.9) -- cycle;
\fill[rotate=135, xshift=2.5, yshift=-3.8]
  (-0.05,0.07) -- (0.05,0.07) -- (0.05,0.9) -- (-0.05,0.9) -- cycle;
\draw[rotate=-45, xshift=-2.6, yshift=1.5,line width=0.04ex*\tkzsymbls@scl, x=0.5ex, y=0.5ex]
(-0.1,0.29) .. controls (-0.7,0.6) and (0,1.2) ..  (0.05,1.7);
\draw[rotate=-45, xshift=-2.1,yshift=1.5,line width=0.04ex*\tkzsymbls@scl, x=0.5ex, y=0.5ex]
(-0.1,0.29) .. controls (0.7,0.6) and (-0.1,1.2) ..  (-0.15,1.7);
\draw[rotate=-45, xshift=-2.5] (-0.1,0.29) .. controls (-0.7,0.6) and (0,1.2) ..  (0,1.5);
\draw[rotate=-45, xshift=-2] (-0.1,0.29) .. controls (0.7,0.6) and (-0.1,1.2) ..  (-0.1,1.5);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Fire#1}%
\tikzsymbolsaftersymbolinput%
}
```

`\Cofeecup` Just a cup of coffee.

```
\ifKV@tikzsymbols@marvosym \else%
\tkzsymbls@Declare@Robust@Command{Coffeecup}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Coffeecup#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{1.82ex*\real{\tkzsymbls@Prmtr}}{1.705ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=0.7ex,y=0.7ex, scale=#1, line width=0.07ex*\tkzsymbls@scl,
decoration={snake,amplitude=.05ex*\tkzsymbls@scl,segment length=0.408ex*\tkzsymbls@scl}]
\draw (0,0) arc (180:270:0.8 and 1) -- ++(0.5,0) arc (270:360:0.8 and 1) -- cycle;
\draw (2.1,-0.15) -- (2.2,-0.15) arc (90:-90:0.3) -- (1.85, -0.75);
\draw[line width=0.05ex*\tkzsymbls@scl, decorate]
   (0.4,0.3) -- +(0,1);
\draw[line width=0.05ex*\tkzsymbls@scl, decorate]
   (1,0.3) -- +(0,1);
\draw[line width=0.05ex*\tkzsymbls@scl, decorate]
   (1.6,0.3) -- +(0,1);
\draw (0,-1.05) -- (2.1,-1.05);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Coffeecup#1}%
\tikzsymbolsaftersymbolinput%
}%
\fi
```

`\Chair` A chair.

```
\tkzsymbls@Declare@Robust@Command{Chair}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Chair#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{0.97ex*\real{\tkzsymbls@Prmtr}}{1.69ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\@tkzssmbls@negtrue\fi%
\begin{tikzpicture}[x=0.9ex,y=0.9ex, scale=#1, line width=0.07ex*\tkzsymbls@scl]
\draw (0,-0.5) -- (0,0.7) -- (0.5,1) -- (0.5,0.25);
\draw[line width=0.06ex*\tkzsymbls@scl] (0,0.4) -- (0.5,0.7);
\draw (0,0) -- (0.5,0.3) -- (1,0) --(1,-0.5);
\if@tkzssmbls@neg\draw (0.5,0.3) -- +(0,-0.5);\fi%
\draw (0.5,-0.3) -- (0.5,-0.8);
\draw (1,0) -- (0.5,-0.3) -- (0,0);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Chair#1}%
\@tkzssmbls@negfalse%
\tikzsymbolsaftersymbolinput%
}
```

`\Bed` A bed.

```
\tkzsymbls@Declare@Robust@Command{Bed}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Bed#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{3.08ex*\real{\tkzsymbls@Prmtr}}{1.68ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=1ex,y=1ex, scale=#1, line width=0.08ex*\tkzsymbls@scl]
\draw (0,0) -- (0,1.6);
\draw (3,0) -- (3,1.2);
\draw (0,0.5) -- (3,0.5);
\draw (0,0.35) -- (3,0.35);
\draw (0.7,0.5) arc (0:90:0.7);
\draw (0.7,0.5) arc(180:30:1.231 and 0.6);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Bed#1}%
\tikzsymbolsaftersymbolinput%
}
```

`\Tribar` Also called Penrose-Triangle

```
\tkzsymbls@Declare@Robust@Commandx{Tribar}[4]
[1=1,2={opacity=0},3={opacity=0},4={opacity=0},usedefault]
{%
```

```
\tikzsymbols@ifsaveboxundefined{Tribar#1#2#3#4}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{1.7175ex*\real{\tkzsymbls@Prmtr}}{1.685ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=0.65ex,y=0.65ex,scale=#1,
   rounded corners=0.03ex*\tkzsymbls@scl, line width=0.06ex*\tkzsymbls@scl]
\fill[#2] (0.15,0.3) -- (-0.15,-0.3) -- (1.75,-0.3) -- ++ (-0.15,-0.3)
   -- (-0.65,-0.6) -- (0.35,1.3) -- +(0.15,-0.3);
\fill[#3] (0,0) -- (1.3,0) -- (0.35,1.9) -- (0.65,1.9) -- (1.75,-0.3) -- (-0.1,-0.3);
\fill[#4]  (1,0) -- (0.35,1.3) --  (-0.65,-0.6) -- ++ (-0.15,0.3) -- (0.35,1.9) -- (1.3,0);
\draw (0,0) -- (1,0) -- (0.5,1) -- cycle;
\draw (0.15,0.3) -- (-0.15,-0.3) -- (1.75,-0.3) -- ++ (-0.15,-0.3)
   -- (-0.65,-0.6) -- (0.35,1.3) -- (0.8,.4);
\draw (0.9,0) -- (1.3,0) -- (0.35,1.9) -- (0.65,1.9) -- (1.75,-0.3) -- +(-0.05,-0.1);
\draw (-0.6,-0.6) -- (-0.65,-0.6) -- ++ (-0.15,0.3) -- (0.35,1.9) -- (0.4,1.9);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Tribar#1#2#3#4}%
\tikzsymbolsaftersymbolinput%
}
```

`\tikzsymbolsMoaithickness` You may already thought it: the line width of the `\Moai`.

```
\newcommand{\tikzsymbolsMoaithickness}{}
```

`\Moai` From the Easter Island: a Moai.

```
\tkzsymbls@Declare@Robust@Commandx{Moai}[1][1=1,usedefault]{%
\tikzsymbols@ifsaveboxundefined{Moai#1}{%
\set@tkzsymblsscl{#1}%
\tksymbls@bx@Prmtrstore{#1}%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\ifdim \tkzsymbls@scl<2pt%
\def\tikzsymbolsMoaithickness{0.05ex}%
\else%
\ifdim \tkzsymbls@scl<5pt%
\def\tikzsymbolsMoaithickness{0.035ex}%
\else%
\def\tikzsymbolsMoaithickness{0.03ex}%
\fi\fi%
\iftikzymbols@draftabsolute%
\tikzsymbols@draftbox{(1.001ex+\tikzsymbolsMoaithickness)*\real{\tkzsymbls@Prmtr}}
    {(1.664ex+\tikzsymbolsMoaithickness)*\real{\tkzsymbls@Prmtr}}%
\else%
\begin{tikzpicture}[x=.13ex, y=.13ex, rounded corners=0.01ex*\tkzsymbls@scl, scale=#1,
    line width=\tikzsymbolsMoaithickness*\tkzsymbls@scl]
\draw  (-2.6,-4.25) --  (-2.5,-5.8)
..controls (-2,-6.8) and (1.5,-6.8) ..  (2.2,-5.8) -- (2.4,-3.95);
```

```
\draw(-2.5,2.5) .. controls (-2.9,4.6) and (2,5) .. (3.3,2.5) -- (2.9,-3.4)
.. controls (2,-5) and (-4,-5) .. (-3.1,-3) -- cycle;
\draw (-2.5,3) -- (-2,5) .. controls (0,6) and (2,5.8) .. (3.1,4.7) -- (3.3,2.5);
\draw[line width=0.02ex*\tkzsymbls@scl]
   (-2.2,-1.8) .. controls (-1,-1.3) and (0,-1.7) .. (1,-2);
\draw[line width=0.02ex*\tkzsymbls@scl]
   (-2.2,-1.8) .. controls (-1,-1) and (0,-1.4) .. (1,-2);
\draw[line width=0.02ex*\tkzsymbls@scl]
   (-2.2,-1.8) .. controls (-1,-2) and (0,-2) .. (1,-2);
\draw (-0.8,4) .. controls (-0.8,3) and (-0.8,2) ..  (-1.6,0.5) -- (-1.8,-0.4)
   .. controls (-1,0.2) and (0,0.2) ..  (0.6,-0.4) -- (0.7,0.4)
   .. controls (0,1) and (0,2) .. (0.8,4);
\draw (-1.8,-0.36) .. controls (-0.5,-0.5) and (0,-0.5) .. (0.6,-0.36);
\draw (3.2,3.5) -- (3.7,3.5) .. controls (3.5,2) and (3.5,2) .. (3.6,-1.5) -- (3,-1.9);
\draw (-2.5,3) .. controls (-2.7,2) and (-3,1) .. (-2.88,-1);
\draw (-2.5,2.8) .. controls (-2,2.5) and (-1,3) .. (-0.8,3.1);
\draw (0.5,3.3) .. controls (1,3) and (1,2.5) .. (3.3,2.4);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Moai#1}%
\tikzsymbolsaftersymbolinput%
}
```

`\Snowman` A snowman. I think its smile is scary.

```
\tkzsymbls@Declare@Robust@Command{Snowman}[1][1]{%
\tikzsymbols@ifsaveboxundefined{Snowman#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{1.545ex*\real{\tkzsymbls@Prmtr}}{1.772ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=0.9ex,y=0.9ex,line width=0.07ex*\tkzsymbls@scl, scale=#1]
\draw (0,0) circle (0.4 and 0.35);
\draw[line width=0.06ex*\tkzsymbls@scl] (0,0.64) circle (0.3 and 0.28);
\draw[line width=0.05ex*\tkzsymbls@scl] (0,1.14) circle (0.2 and 0.2);
\draw[rounded corners=0.1ex*\tkzsymbls@scl,line width=0.05ex*\tkzsymbls@scl,
   rotate around={-30:(0,1.14)}]
   (-0.2,1.15) -- ++(0,0.35) -- +(0.4,0) -- (0.2,1.14);
\draw[rounded corners=0.07ex*\tkzsymbls@scl,line width=0.05ex*\tkzsymbls@scl,
    rotate around={-30:(0,1.14)}]
    (-0.2,1.19) arc (270:90:0.1);
\fill (0,0.78) circle (0.04);
\fill (0,0.63) circle (0.04);
\fill (0,0.48) circle (0.04);
\fill (0,0.2) circle (0.05);
\fill (0,0) circle (0.05);
\fill (0,-0.2) circle (0.05);
\fill (-0.06,1.18) circle (0.045);
```

```
\fill (0.06,1.18) circle (0.045);
\fill (0.1,1.08) circle (0.015);
\fill (-0.1,1.08) circle (0.015);
\fill (0.06,1.055) circle (0.015);
\fill (-0.06,1.055) circle (0.015);
\fill (0.02,1.039) circle (0.015);
\fill (-0.02,1.039) circle (0.015);
\draw (-0.3,0.7) -- (-0.6,0.8);
\draw (-0.6,0.8) -- (-0.75,0.7);
\draw (-0.6,0.8) -- (-0.55,1);
\draw (-0.6,0.8) -- (-0.8,0.9);
\draw[line width=0.06ex*\tkzsymbls@scl] (-0.65,0) -- (-0.65,1);
\foreach\x in {-0.85, -0.75,-0.65,-0.55,-0.45}
\draw[line width=0.05ex*\tkzsymbls@scl] (-0.65,1) -- (\x,1.3);
\draw (0.3,0.7) -- (0.6,0.8);
\draw (0.6,0.8) -- (0.75,0.7);
\draw (0.6,0.8) -- (0.6,1);
\draw (0.6,0.8) -- (0.8,0.9);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{Snowman#1}%
\tikzsymbolsaftersymbolinput%
}
```

6.4 Trees

Many great ideas are stolen. Don't know who said that, but it's true.

`\BasicTree` We define our `\BasicTree`. We check if the last parameter is "leaf", if not we check if the last parameter is empty, if not: we generate an error message:

```
\tkzsymblsnewcommand{BasicTree}[5][1]{%
\tikzsymbols@ifsaveboxundefined{BasicTree#1#2#3#4#5}{%
\def\leaf@or@not@leaf{#5}%
\ifx\leaf@or@not@leaf\@leaf@is@leaf%
\Basic@Tree[#1]{#2}{#3}{#4}{#5}%
\else%
\ifx\\#5\\%
\Basic@Tree[#1]{#2}{#3}{#4}{#5}%
\else%
\PackageError{tikzsymbols}{The last\MessageBreak parameter has either to be \MessageBreak
'leaf' or has to be empty}{See the tikzsymbols documentation. Section 'Trees'.}%
\fi\fi%
}%
\tikzsymbols@use@box{BasicTree#1#2#3#4#5}%
\tikzsymbolsaftersymbolinput%
}
```

`\WorstTree` It's not that bad.

```
\tkzsymbls@Declare@Robust@Command{WorstTree}[1][1]{%
\tikzsymbols@ifsaveboxundefined{WorstTree#1}{%
\set@tkzsymblsscl{#1}%
\iftikzymbols@draftabsolute%
\tksymbls@bx@Prmtrstore{#1}%
\tikzsymbols@draftbox{1.64ex*\real{\tkzsymbls@Prmtr}}{1.84ex*\real{\tkzsymbls@Prmtr}}%
\else%
\ifdim\tkzsymbls@scl<0pt\set@tkzsymblsscl{-#1}\fi%
\begin{tikzpicture}[x=1ex,y=1ex, line width=0.04ex*\tkzsymbls@scl,scale=#1]
\fill[brown] (-0.3,0) .. controls (0.2,0.3) and (0.2,0.7) .. (0.2,1) -- (0.5,1) ..
   controls (0.5,0.7) and (0.5,0.3) .. (1,0);
\draw (-0.3,0) .. controls (0.2,0.3) and (0.2,0.7) .. (0.2,1) -- (0.5,1) ..
    controls (0.5,0.7) and (0.5,0.3) .. (1,0) ;
\fill[green] (0.2,0.8) --  (0,0.8) .. controls (-0.4,0.7) and (-0.4,1) ..  (-0.3,1.2) ..
    controls (-0.3, 1.6) and (-0.1,1.6) .. (0.1,1.5) ..
    controls (0.3,1.8) and (0.6,1.6) ..  (0.7,1.5) ..
    controls (1.1, 1.6) and (1,1.4) ..  (1,1.2) ..
    controls (1.2,1) and (1.2,0.7) .. (0.8,0.8) -- (0.5,0.8);
\draw (0.214,0.8) --  (0,0.8) .. controls (-0.4,0.7) and (-0.4,1) ..  (-0.3,1.2) ..
   controls (-0.3, 1.6) and (-0.1,1.6) .. (0.1,1.5) ..
   controls (0.3,1.8) and (0.6,1.6) ..  (0.7,1.5) .. controls (1.1, 1.6) and (1,1.4) ..
   (1,1.2) .. controls (1.2,1) and (1.2,0.7) .. (0.8,0.8) -- (0.486,0.8);
\fill[red] (0,1) circle (0.1);
\fill[red] (0.4,1.2) circle (0.1);
\fill[red] (0.8,1.1) circle (0.1);
\end{tikzpicture}%
\fi%
}%
\tikzsymbols@use@box{WorstTree#1}%
\tikzsymbolsaftersymbolinput%
}
```

`\Springtree` `\Summertree` `\Autumntree` `\Wintertree`

Some predefined Trees.

"Hey that look like the trees in the ..." – "Yes, Yes, I know!".

We don't need `\tikzsymbolsaftersymbolinput` because it is already used in `\BasicTree`.

```
\tkzsymbls@Declare@Robust@Commandx{Springtree}[1][1=1, usedefault]%
  {\tikzsymbolsuse{BasicTree}[#1]{brown!70!black}{green!90!black}{green!80!black}{leaf}}
\tkzsymbls@Declare@Robust@Commandx{Summertree}[1][1=1, usedefault]%
  {\tikzsymbolsuse{BasicTree}[#1]{brown!50!black}{green!80!black}{red!80!green}{leaf}}
\tkzsymbls@Declare@Robust@Commandx{Autumntree}[1][1=1, usedefault]%
  {\tikzsymbolsuse{BasicTree}[#1]{red!30!black}{red!75!black}{orange}{leaf}}
\tkzsymbls@Declare@Robust@Commandx{Wintertree}[1][1=1, usedefault]%
  {\tikzsymbolsuse{BasicTree}[#1]{black!80!}{black!50}{black!25}{}}
```

`Error Message`

If option `marvosym` is active, but the package not loaded, there will be an error message.

```
\AtBeginDocument{%
\ifKV@tikzsymbols@marvosym
```

```
\@ifpackageloaded{marvosym}{}{%
\PackageError{tikzsymbols}{Use option 'marvosym' only\MessageBreak
if you load package 'marvosym'}
{Either load package 'marvosym' or\MessageBreak
delete the tikzsymbols option 'marvosym'}}
\fi
}
```

Change History

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.

O

P

R

S

T

U

V

W

Well then, happy TEXing!