The TikZ-Extensions Package
Manual for version 0.5.1
https://github.com/Qrrrbirlbel/tikz-extensions
Qrrrbirlbel
April 2, 2023

Contents

I Introduction 5
  1 Usage 5
  2 Why do we need it? 5
  3 Having problems? 5

II TikZ Libraries 6
  4 Calendar 7
    4.1 Value-keys and nestable if key 7
    4.2 PGFmath functions 7
    4.3 Week numbering (ISO 8601) 7
  5 Layers 8
    5.1 Internal keys 8
    5.2 User-level keys 8
6 Node Families
   6.1 Text Box ................................................................. 9
   6.2 Minimum Width/Height ............................................. 10
   6.3 More shapes that support the keys width and height .......... 11

7 Nodes
   7.1 Pic as a node .......................................................... 12
   7.2 Nodes on paths ....................................................... 12
      7.2.1 Nodes on Lines .................................................. 12
      7.2.2 Nodes on Curves ................................................ 13

8 Arc to a point ................................................................... 14

9 More Horizontal and Vertical Lines
   9.1 Zig-Zag ................................................................. 16
   9.2 Zig-Zig ............................................................... 18
   9.3 Even more Horizontal and Vertical Lines ....................... 19

10 Extending the Path Timers ............................................. 21
    10.1 Rectangle ............................................................. 21
    10.2 Parabola ............................................................ 22
    10.3 Sine/Cosine ......................................................... 22

11 Using Images as a Pattern ............................................. 24

12 Positioning Plus .......................................................... 25
    12.1 Useful corner anchors .............................................. 25
    12.2 Useful placement keys for vertical and horizontal alignment .................................................. 26

13 Scaling Pictures to a Specific Size .................................. 30
    13.1 Keeping the aspect ratio .......................................... 30
    13.2 Changing the aspect ratio ....................................... 30

14 Arcs through Three Points ............................................ 32

15 Mirror, Mirror on the Wall ............................................ 33
    15.1 Using the reflection matrix ...................................... 33
    15.2 Using built-in transformations ................................ 34
Part I

Introduction

1 Usage

This package is called `tikz-ext`, however, one can’t load it via `\usepackage`. Instead, this package consists mostly of PGF and TiZ libraries which are loaded by either `\usepgflibrary` or `\usetikzlibrary`.

2 Why do we need it?

Since I have been answering questions on TeX.sx I’ve noticed that some questions come up again and again, every time with a slightly different approach on how to solve them.

I don’t like reinventing the wheel which is why I’ve gathered the solutions of my answers in this package.

3 Having problems?

Note however, that most of these extensions haven’t been stress-tested properly and might be considered experimental.

Don’t hesitate to open an issue on GitHub. You probably found a bug.

---

1Except for `calendar-ext` and `pgffor-ext`. 
Part II

TikZ Libraries

These libraries only work with TikZ.
4 Calendar

TikZ Library `ext.calendar-plus`

\usetikzlibrary{ext.calendar-plus} \% \LaTeX{} and plain \TeX{}
\usetikzlibrary{ext.calendar-plus} \% Con\TeX{}

This library extends the TikZ library `calendar`.

Q & A: \[10, 11, 4\] & \[26, 46, 44\]

4.1 Value-keys and nestable if key

The values of following keys are originally stored in some macros that are not accessible by the user. These are now simple value-keys. The `@`-protected macros are still available, of course.

/tikz/day xshift (initially 3ex)
/tikz/day yshift (initially 3.5ex)
/tikz/month xshift (initially 9ex)
/tikz/month yshift (initially 9ex)

It is now also possible to nest \texttt{/tikz/if} occurrences.

/tikz/if=(⟨conditions⟩)(⟨code or options⟩)\texttt{else}⟨⟨else code or options⟩⟩ (no default)

4.2 PGFMATH functions

\texttt{weeksinmonthofyear} (first weekday, month, year)
\texttt{\pgfmathweeksinmonthofyear} (first weekday)⟨month⟩⟨year⟩

Returns the number of (partial) weeks in the month \texttt{month} of year \texttt{year} when this month begins on a \texttt{first weekday}.

\texttt{lastdayinmonthofyear} (month, year)
\texttt{\pgfmathlastdayinmonthofyear} (month)⟨year⟩

Returns the last day (28, 29, 30 or 31) of month \texttt{month} of year \texttt{year}.

4.3 Week numbering (ISO 8601)

The actual week number algorithm is implemented by the \texttt{pgfcalendar-ext} package/module in section 24.2.

\texttt{/tikz/week code=⟨code⟩} (no default)

Works like \texttt{/tikz/day code} or \texttt{/tikz/month code}, only for weeks.

\texttt{/tikz/week text=⟨text⟩} (no default)

Works like \texttt{/tikz/day text} or \texttt{/tikz/month text}, only for weeks.

\texttt{/tikz/every week} (style, no value)

Works like \texttt{/tikz/every day} or \texttt{/tikz/every month}, only for weeks.

\texttt{/tikz/week label left} (style, no value)

Places the week label to the left of the first day of the month. (For \texttt{week list} and \texttt{month list} where a week does not start on a Monday, the position is chosen “as if” the week had started on a Monday – which is usually exactly what you want.)
5 Layers

TikZ Library `ext.layers`

`\usetikzlibrary{ext.layers}` % LaTeX and plain \TeX
`\usetikzlibrary{ext.layers}` % Con\TeX

This library extends TikZ's functionalities to put nodes, edges, matrices and pics on a separate layer without having to use the `pgfonlayer` environment.

Consider this library experimental. If you can, avoid it and use the `pgfonlayer` environment or change the drawing order.

5.1 Internal keys

`/tikz-ext/patch={specification}` (no default)

Since this library is experimental, its functionality needs to be activated explicitly. The `{specification}` is one of

- node,
- matrix,
- pic\(^2\),
- edge or
- all which applies all the patches at once.

These keys only work when a patch is applied but don’t need to be used since the patching activated specific

`/tikz-ext/layers/in box={box}` (no default)
`/tikz-ext/layers/on layer={layer}` (no default)

5.2 User-level keys

`/tikz/node on layer={layer}` (no default)
`/tikz/node in box={box}` (no default)

`/tikz/matrix on layer={layer}` (no default)
`/tikz/matrix in box={box}` (no default)

`/tikz/edge on layer={layer}` (no default)
`/tikz/edge in box={box}` (no default)

`/tikz/pic on layer={layer}` (no default)
`/tikz/pic in box={box}` (no default)

\(^2\)Only the normal `/tikz/pics/code` can be placed on different layers. Both `/tikz/pics/background code` and `/tikz/pics/foreground code` will not be affected.
6 Node Families

TikZ Library \texttt{ext.node-families}

\begin{verbatim}
\usetikzlibrary{ext.node-families} \% \LaTeXX and plain TeX
\usetikzlibrary{ext.node-families} \% Con\TeX\n\end{verbatim}

With this library the user can instruct multiple nodes to have the same width, height, text width, text height or text width. This uses the hook \texttt{/tikz/execute at end picture} to write the nodes’ measurements to the \texttt{aux} file.

Unfortunately, this does not work with the external library.\footnote{First of all, I can’t figure out how to use the \texttt{aux} file during externalization since it gets written to the \texttt{log} instead. And then there’s the question about how \texttt{external} would notice the need to export the picture again until it’s stable...}

\textbf{Q \& A: \cite{13} \& \cite{29}}

Before we get to the interesting keys, a common prefix can be set for the families’ names. Initially this is \texttt{\pgfpictureid-} so that families of different pictures don’t interact.

\begin{verbatim}
/tikz/node family/prefix=⟨prefix⟩ (no default, initially \texttt{\pgfpictureid-})
\end{verbatim}

The family names are prefixed with the value of \texttt{/tikz/node family/prefix}.

6.1 Text Box

The following keys – when setup, see below – work with every shape with one single node part.\footnote{Technically, it will also work with shapes with multiple node parts but it will only affect the main node part.} Initially though, only \texttt{circle} and \texttt{rectangle} are set up that way.

\begin{verbatim}
/tikz/node family/text height=⟨name⟩ (no default, initially \{\})
\end{verbatim}

Nodes with the same \texttt{⟨name⟩} will have the same text height. An empty \texttt{⟨name⟩} disables the evaluation by the library.

\begin{verbatim}
/tikz/node family/text depth=⟨name⟩ (no default, initially \{\})
\end{verbatim}

Nodes with the same \texttt{⟨name⟩} will have the same text depth. An empty \texttt{⟨name⟩} disables the evaluation by the library.

\begin{verbatim}
/tikz/node family/text width=⟨name⟩ (no default, initially \{\})
\end{verbatim}

Nodes with the same \texttt{⟨name⟩} will have the same text width. An empty \texttt{⟨name⟩} disables the evaluation by the library.

\begin{verbatim}
/tikz/node family/text=⟨name⟩ (no default)
\end{verbatim}

Sets text height, text depth and text width.

Since the width of the node’s content’s box is setup much earlier, the previous key only extends the width of that box which would make the text seem as if it were aligned to the left. With \texttt{text width family align} this can changed.
/tikz/node family/text width align=(alignment)  
(alignment) is one of left, center or right.

/foo {alignment} (no default, initially center)

\usetikzlibrary{positioning,ext.node-families}
\tikzexternaldisable % ext.node-families does not work with active externalization
\begin{tikzpicture}[nodes={rectangle, draw, node family={text width=manual, text width align=right}}]
\node (a) {Foo};
\node[below=of a] (b) {Foobar};
\end{tikzpicture}

/tikz/node family/setup shape=(shape)  
(no default)

This adds instructions to the (shape)’s definition which adjust the text box’s dimensions according to the family.
This should be only used once per shape.

6.2 Minimum Width/Height

While the keys of the previous subsection work well enough for nodes of the same shape (and the same inner seps), for different node shapes the text box dimensions will be used differently for the node’s total dimension.

For this, the following keys are necessary. When one of the keys are used the values of minimum width and/or minimum height are set to nf_width or nf_height respectively.

/tikz/node family/width=(name)  
(no default, initially {})

Nodes with the same (name) will have the same \texttt{/pgf/minimum width}. An empty (name) disables the evaluation by the library.

/foo {name} (no default, initially {})

\usetikzlibrary{positioning,ext.node-families}
\tikzexternaldisable % ext.node-families does not work with active externalization
\begin{tikzpicture}[nodes={rectangle, draw, node family/width=manual}]
\node (a) {Foo};
\node[below=of a] (b) {Foobar};
\end{tikzpicture}

/tikz/node family/height=(name)  
(no default, initially {})

Nodes with the same (name) will have the same \texttt{/pgf/minimum height}. An empty (name) disables the evaluation by the library.

/foo {name} (no default)

\usetikzlibrary{positioning,ext.node-families}
\tikzexternaldisable % ext.node-families does not work with active externalization
\begin{tikzpicture}[nodes={rectangle, draw, node family/size=manual}]
\node (a) {Foo};
\node[below=of a] (b) {Foobar};
\end{tikzpicture}

/tikz/node family/size=(name)  
(set both height and width.)  
(no default)

/
6.3 More shapes that support the keys width and height

**TikZ Library** `ext.node-families.shapes.geometric`

\begin{verbatim}
\usetikzlibrary{ext.node-families.shapes.geometric} % \LaTeX and plain \TeX
\usetikzlibrary{ext.node-families.shapes.geometric} % Con\TeX

This library adds support for the keys `/tikz/node family/width` and `/tikz/node family/height` for the shapes of the PGF library `shapes.geometric`.

Q: [21]

The shapes are also setup for the keys from subsection 6.1.
\end{verbatim}
7 Nodes

TIkZ Library `ext.nodes`

\usetikzlibrary{ext.nodes} \% \LaTeX\ and plain \TeX\  
\usetikzlibrary{ext.nodes} \% Con\TeX\  

This library extends Ti\kZ's functionalities when it comes to nodes.

Q & A: [9, 17] & [30, 39]

7.1 Pic as a node

/tikz/pic=⟨boolean⟩  
(default true, initially false)

This key allows one to use a pic where usually only nodes are accepted, for example as a label.

7.2 Nodes on paths

When nodes are placed along paths they don’t interrupt the path at that place. The decoration markings and its \texttt{pgf/decoration/mark connection node} key can help but only works for straight paths and doesn’t play nicely with arrow tips.

This library provides alternatives. These are separated into straight paths, i.e. \texttt{--}, and everything else (including any \texttt{to path}).

7.2.1 Nodes on Lines

/tikz/node on line=⟨anchor specification⟩  
(style, default {})  
This installs a \texttt{/tikz/to path} that places one node along a straight line but connect the line with it.

This allows a node to be placed on a straight line without having to use \texttt{fill = white} or similar tricks to make the line disappear beneath the node.

The optional \texttt{(anchor specification)} allows to specify the anchors to which the line should connect. It allows one or two anchors divided by \texttt{and} to be specified.

/tikz/nodes on line  
(style, no value)

This is similar to the previous key but allows multiple nodes to be placed on a straight line if they are in the correct order (from start to target), don’t overlap with each other, the start or the target.

It allows no anchor specification.
7.2.2 Nodes on Curves

The following keys need the \texttt{intersections} and the \texttt{spath3} \cite{spath3} library to be loaded. They will not be automatically loaded by this library.

Any \texttt{/pgf/outer sep} will be ignored.

If you can, use \texttt{fill\\{(bg color)\}} instead of these keys, it will be much faster and easier.

\texttt{/tikz/nodes on curve=(to path)} \hfill (style, default line to)

Similar to \texttt{/tikz/nodes on line}, this key allows to have nodes on arbitrary paths.

This is not suitable for paths connecting nodes.

\texttt{/tikz/nodes on curve'=\{(to path)\}} \hfill (style, default line to)

As above but suitable for connecting nodes.
8 Arc to a point

TikZ Library ext.paths.arcto

\usetikzlibrary{ext.paths.arcto} \% \LaTeX{} and plain \TeX{}
\usetikzlibrary{ext.paths.arcto} \% Con\TeX

This library adds the new path operation \texttt{arc to} that specifies an arc \texttt{to} a point – without the user having to specify any angles.

\begin{tikzpicture}[ultra thick,dot/.style={label=\tiny{#1}}]
\coordinate[dot=below left:$a$] (a) at (0,0);
\coordinate[dot=above right:$b$] (b) at (2,3);
\begin{scope}[radius=3, nodes={shape=circle, fill=white, fill opacity=.9, text opacity=1, inner sep=+0pt, sloped, allow upside down}]
\draw[blue] (a) arc to node[near start] {.25} node {.5} node[near end] {.75} (b);
\draw[red] (a) arc to[clockwise] node[near start] {.25} node {.5} node[near end] {.75} (b);
\draw[blue!50] (a) arc to[large] node[near start] {.25} node {.5} node[near end] {.75} (b);
\draw[red!50] (a) arc to[large, clockwise] node[near start] {.25} node {.5} node[near end] {.75} (b);
\end{scope}
\fill[radius=2pt] (a) circle[] (b) circle[];
\end{tikzpicture}

```
\path ... \texttt{arc to}((options))(coordinate or cycle) ...;
```

When this operation is used, the path gets extended by an arc that goes through the current point and \texttt{(coordinate)}.

For two points there exist two circles or four arcs that go through or connect these two points. Which one of these is constructed is determined by the following options that can be used inside of \texttt{(options)}.

```
/tikz/arc to/clockwise
```

(style, no value)
This constructs an arc that goes clockwise.

\tikz/arc to/counter clockwise
  This constructs an arc that goes counter clockwise.
  This is the default.

\tikz/arc to/large
  This constructs an arc whose angle is larger than 180°.

\tikz/arc to/small
  This constructs an arc whose angle is smaller than 180°.

\tikz/arc to/rotate=(degree)
  Rotates the arc by (degree). This is only noticeable when \textit{x radius} and \textit{y radius} are different.

\tikz/arc to/x radius=(value)
  This forwards the (value) to \textit{tikz/x radius}. Its (value) is used for the radius of the arc.

\tikz/arc to/y radius=(value)
  This forwards the (value) to \textit{tikz/y radius}. Its (value) is used for the radius of the arc.

\tikz/arc to/radius=(value)
  This forwards the (value) to both \textit{tikz/x radius} and \textit{tikz/y radius}. Its (value) is used for radius of the arc.

\tikz/every arc to
  After \textit{tikz/every arc} this will also be applied before any (options) are set.

It should be noted that this uses \texttt{\pgfpatharcto} for which the TikZ manual warns:

\begin{quote}
The internal computations necessary for this command are numerically very unstable. In particular, the arc will not always really end at the (target coordinate), but may be off by up to several points. A more precise positioning is currently infeasible due to \textsc{tikz}'s numerical weaknesses. The only case it works quite nicely is when the resulting angle is a multiple of 90°.
\end{quote}

The arc to path operation will also work only in the canvas coordinate system. The lengths of the vectors (1, 0) and (0, 1) will be used for the calculation of the radii but no further consideration is done.
9 More Horizontal and Vertical Lines

TikZ Library `ext.paths.ortho`

\usetikzlibrary{ext.paths.ortho} % \LaTeX{} and plain \TeX{}
\usetikzlibrary{ext.paths.ortho} % Con\TeX

This library adds new path specifications `-|`, `|-` as well as `r-ud`, `r-du`, `r-rl` and `r-rr`.

9.1 Zig-Zag

Similar to the path operations `-| and `-|` this library adds the path operations `-|` and `-|`.

\path \ldots `-|` [(\textit{options})](\textit{coordinate or cycle}) \ldots;

This operation means “first vertical, then horizontal and then vertical again”.

\path \ldots `-|` [(\textit{options})](\textit{coordinate or cycle}) \ldots;

This operation means “first horizontal, then vertical and then horizontal again”.

\texttt{/tikz/ortho/ratio=}(\textit{ratio})

\begin{tikzpicture}[very thick, rounded corners]
\draw [help lines] (-.25, -1.25) grid (2.25, 1.25);
\draw (0, 0) \ldots `-|` [(\textit{options})](\textit{coordinate or cycle}) \ldots;
\end{tikzpicture}

\texttt{/tikz/ortho/distance=}(\textit{distance})

This sets the distance between the start point and the middle part of the Zig-Zag connection.

\begin{tikzpicture}[very thick, rounded corners]
\draw [help lines] (-.25, -1.25) grid (2.25, 1.25);
\draw (0, 0) \ldots `-|` [(\textit{options})](\textit{coordinate or cycle}) \ldots;
\end{tikzpicture}

\texttt{/tikz/ortho/distance=}(\textit{distance})

This sets the distance between the start point and the middle part of the Zig-Zag connection.

\begin{tikzpicture}[very thick, rounded corners]
\draw [help lines] (-.25, -1.25) grid (2.25, 1.25);
\draw (0, 0) \ldots `-|` [\texttt{/tikz/ortho/distance=}.25](\textit{coordinate or cycle}) \ldots;
\end{tikzpicture}
\usetikzlibrary{ext.paths.ortho}
\begin{tikzpicture}[very thick,-latex]
draw[help lines,-](-.25, -.25) grid (5.25, 3.25);
draw (0, 0) |- (distance=.5cm) ++(2, 1);
draw (0, 2) |- (distance=-.5cm) ++(2, 1);
tikzset{xshift=3cm}
draw (2, 1) |- (distance=.5cm) ++(-2, -1);
draw (2, 3) |- (distance=-.5cm) ++(-2, -1);
\end{tikzpicture}

\texttt{/tikz/ortho/from center=⟨true or false⟩} (default \texttt{true})

When nodes get connected the placement of the middle part of the Zig-Zag and the Zig-Zig (see below) connections will be calculated from the border of these nodes. The middle part of the connections can be calculated from the nodes' center if this key is set to \texttt{true}.

New timers are setup for both the Zig-Zag and the Zig-Zig connections, these can be configured through the following keys.

\usetikzlibrary{paths.ortho}
\tikz\draw (0,0) -| (2,3)
\foreach \p in {0.0, 0.25, 0.5, 0.75, 1.0}{
node [pos=\p] {\p};
}

\texttt{/tikz/ortho/spacing=⟨number⟩} (no default, initially 4)

Unless \texttt{⟨number⟩} = 0 is set

- \texttt{pos = 0} will be at the start,
- \texttt{pos = 1} will be at the end,
- \texttt{pos = \frac{1}{⟨number⟩}} will be at the first kink,
- \texttt{pos = \frac{⟨number⟩-1}{⟨number⟩}} will be at the second kink and
- \texttt{pos = .5} will be in the middle of the middle part of the connection.
If \( \langle \text{number} \rangle = 0 \) then

- \( \text{pos} = -1 \) will be at the start,
- \( \text{pos} = 2 \) will be at the end,
- \( \text{pos} = 0 \) will be at the first kink,
- \( \text{pos} = 1 \) will be at the second kink and
- \( \text{pos} = .5 \) will still be in the middle of the middle part of the connection.

\texttt{/tikz/ortho/middle 0 to 1}

This is an alias for \( \text{spacing} = 0 \).

### 9.2 Zig-Zig

\texttt{\path ... r-ud\langle\textit{options}\rangle\langle\textit{coordinate or cycle}\rangle \ldots;}

This operation means "first up, then horizontal and then down".

\texttt{\texttt{/tikz/ortho/ud distance=\langle\textit{length}\rangle}}

This sets the distance between the start and the horizontal line to \( \langle \textit{length} \rangle \).

\texttt{\path ... r-du\langle\textit{options}\rangle\langle\textit{coordinate or cycle}\rangle \ldots;}

This operation means "first down, then horizontal and then up".

\texttt{\texttt{/tikz/ortho/du distance=\langle\textit{length}\rangle}}

This sets the distance between the start and the horizontal line to \( \langle \textit{length} \rangle \).

\texttt{\path ... r-lr\langle\textit{options}\rangle\langle\textit{coordinate or cycle}\rangle \ldots;}

This operation means "left down, then vertical and then right".

\texttt{\texttt{/tikz/ortho/lr distance=\langle\textit{length}\rangle}}

This sets the distance between the start and the vertical line to \( \langle \textit{length} \rangle \).

\texttt{\path ... r-rl\langle\textit{options}\rangle\langle\textit{coordinate or cycle}\rangle \ldots;}

This operation means "first right, then vertical and then down".

\texttt{\texttt{/tikz/ortho/rl distance=\langle\textit{length}\rangle}}

This sets the distance between the start and the vertical line to \( \langle \textit{length} \rangle \).

All distances can be set with one key.

\texttt{\texttt{/tikz/ortho/udlr distance=\langle\textit{length}\rangle}}

Sets all the previous distances to the same value \( \langle \textit{length} \rangle \).
9.3 Even more Horizontal and Vertical Lines

The following keys can be used to access vertical and horizontal line path operations.

/tikz/horizontal vertical
This installs to path = -| (\tikztotarget) \tikztonodes that can be used with the path operations to or edge.

/tikz/vertical horizontal
This installs to path = |- (\tikztotarget) \tikztonodes that can be used with the path operations to or edge.

/tikz/horizontal vertical horizontal
This installs to path = -|- (\tikztotarget) \tikztonodes that can be used with the path operations to or edge.

/tikz/vertical horizontal vertical
This installs to path = |-| (\tikztotarget) \tikztonodes that can be used with the path operations to or edge.

When connecting rectangular nodes, these keys could be useful as well. They all need to be given to a to or edge path operation.

/tikz/only vertical second=(length)
This draws a vertical line from the start point to the target point so that it connects to the target point in the center (or at its border in case it is a node).

The optional (length) can be used to shift the line orthogonally to its direction.

/tikz/only horizontal second=(length)
This draws a horizontal line from the start point to the target point so that it connects to the target point in the center (or at its border in case it is a node).

The optional (length) can be used to shift the line orthogonally to its direction.

/tikz/only vertical first=(length)
This draws a vertical line from the start point to the target point so that it connects to the start point in the center (or at its border in case it is a node).

The optional (length) can be used to shift the line orthogonally to its direction.

/tikz/only horizontal first=(length)
This draws a horizontal line from the start point to the target point so that it connects to the start point in the center (or at its border in case it is a node).

The optional (length) can be used to shift the line orthogonally to its direction.
Since all previous keys are rather cumbersome, one can install shortcuts for these.

\texttt{/tikz/ortho/install shortcuts} \hspace{1cm} \texttt{(style, no value)}

Installs the following shortcuts:
\begin{itemize}
  \item [-] \rightarrow \text{vertical horizontal}
  \item [-|] \rightarrow \text{horizontal vertical}
  \item [-|-] \rightarrow \text{horizontal vertical horizontal}
  \item [|-] \rightarrow \text{vertical horizontal vertical}
  \item [**] \rightarrow \text{only vertical first}
  \item [*[ ] \rightarrow \text{only vertical second}
  \item [-*] \rightarrow \text{only horizontal first}
  \item [*-] \rightarrow \text{only horizontal second}
\end{itemize}
10 Extending the Path Timers

TikZ Library `ext.paths.timer`

\usetikzlibrary{ext.paths.timer} % \LaTeX and plain \TeX
\usetikzlibrary{ext.paths.timer} % Con\TeX

This library adds timers to the path specifications rectangle, parabola, sin and cos.

Q & A: [6, 5] & [36, 48]

In TikZ, the path specification rectangle, parabola, sin and cos do not provide their own timer, i.e. a node placing algorithm that is dependent on the actual path. For rectangle the timer of the straight line between the rectangle’s corners is used, for the other paths, nodes, coordinates, pics, etc. are placed on the last coordinate.

This library allows this.

10.1 Rectangle

For the rectangle path operator, the timer starts with pos = 0 (= at start) from the starting coordinate in a counter-clockwise direction along the rectangle. The corners will be at positions 0.0, 0.25, 0.5, 0.75 and 1.0.

/tikz/rectangle timer=line or rectangle

By default, the library activates the new (correct) timer for rectangle. With rectangle timer = line the original line timer can be reinstated.
10.2 Parabola

For the parabola path operator the timer is similar to the \texttt{.. controls ..} operator.

The position 0.5 will lie at the bend.

\begin{tikzpicture}
\draw[help lines] (-2.25, -1.25) grid (2.25, 3.25);
\draw (2,-1) parabola bend (0,0) (-1,3);
\draw[ultra thick] (-2,-1) parabola bend (0,0) (1,3)
foreach \pos in {1,...,4,6,7,...,9}{
  node[pos=\pos, sloped, fill=white, font=\small, inner sep=+0pt]
  {\pos};
}\end{tikzpicture}

If no bend is specified half the positions will collapse into one end of the curve.

\begin{tikzpicture}[every pin edge/.style={latex-, shorten <=1pt, gray}]
\draw (-2,-2) parabola (1,0)
foreach \pos in {0, 1, ..., 10} {
  node [pos=\pos/10, pin={[anchor=-18*\pos+90]-18*\pos+270:\pos}];
}\end{tikzpicture}

10.3 Sine/Cosine

The \texttt{sin} and \texttt{cos} path operators also allow placing of nodes along their paths.

\begin{tikzpicture}[every pin edge/.style={latex-, shorten <=1pt, gray}]
\begin{scope}[xshift=-1cm]
\draw (-2,-2) parabola (1,0)
foreach \pos in {0, 1, ..., 10} {
  node [pos=\pos/10, pin={[anchor=-18*\pos+90]-18*\pos+270:\pos}];
}\end{scope}
\end{tikzpicture}
\usetikzlibrary{ext.paths.timer}
\begin{tikzpicture}[mark nodes on line/.style={insert path={
  \foreach \pos in {1, ..., 9} {node[sloped, fill=white, font=\small, inner sep=+0pt, pos=\pos/10] {\pos}}}}]
  \draw[help lines] (-2.1,-2.1) grid (2.1,0.1);
  \draw (-2,-2) sin (1,0) [mark nodes on line];
  \draw[shift=(0:1)](-2,-2) cos (1,0) [mark nodes on line];
\end{tikzpicture}
11 Using Images as a Pattern

TikZ Library `ext.patterns.images`

\usetikzlibrary{ext.patterns.images} % \LaTeX and plain \TeX
\usetikzlibrary{ext.patterns.images} % Con\TeX

This library allows to use an image to be used as a repeating pattern for a path.

Q & A: [16] & [47]

With this library arbitrary images (or indeed PDF documents) can be used as a repeating pattern for the background of a path. This is a two-step process:

1. Declaring an image as an “image-pattern”.
2. Using the “image-pattern”.

\pgfsetupimageaspattern\{(options)\}\{(name)\}\{(image)\}

\tikz\{image as pattern\{(options)\}\}

/default ()

\tikz\{/tikz/image as pattern\{(options)\}\}

\pgfsetupimageaspattern\[width=.5cm\]{grid}\{example-image-1x1\}
\tikz \node[star, minimum size=3cm, draw, image as pattern={name=grid, options={left, bottom, y=+.5cm, rotate=45}}] {;}

\tikz\{/tikz/image as pattern\{(options)\}\}\{(name)\}

Specifies the name of the “image-pattern” to be used.

\tikz\{/tikz/image as pattern\{(options)\}\}\{(option)\}

Options that will be used by the internal \pgf\text, only keys from \pgf\text should be used.

\tikz\{/tikz/image as pattern\{(options)\}\}\{(style)\}

Appends style /tikz/image as pattern\{(options)\}.
12 Positioning Plus

TikZ Library `ext.positioning-plus`

\usepackage[ext.positioning-plus]{tikz}

With the help of the positioning and the fit library this extends the placement of nodes.

12.1 Useful corner anchors

The anchors `corner north east`, `corner north west`, `corner south west` and `corner south east` are defined as "generic anchors", i.e. they are defined for all shapes. This is mostly useful for the placement of circular shapes.

- `/tikz/corner above left=⟨specification⟩` (style, default 0pt)
  
  Similar as `/tikz/above left` of the TikZ library positioning but uses the `corner north west` anchor.

- `/tikz/corner below left=⟨specification⟩` (style, default 0pt)
  
  Similar as `/tikz/below left` of the TikZ library positioning but uses the `corner south west` anchor.

- `/tikz/corner above right=⟨specification⟩` (style, default 0pt)
  
  Similar as `/tikz/above right` of the TikZ library positioning but uses the `corner north east` anchor.

- `/tikz/corner below right=⟨specification⟩` (style, default 0pt)
  
  Similar as `/tikz/below right` of the TikZ library positioning but uses the `corner south east` anchor.
12.2 Useful placement keys for vertical and horizontal alignment

\textit{/tikz/north left=⟨specification⟩}

Like /tikz/left but aligns the nodes at their north border.

This is basically the same as \texttt{left=of reference.north west, anchor=north east}. 
/tikz/north right=(specification)  
Like /tikz/right but aligns the nodes at their north border. 
This is basically the same as left=of reference.north east, anchor=north west.

/tikz/south left=(specification)  
Like /tikz/left but aligns the nodes at their south border. 
This is basically the same as left=of reference.south west, anchor=south east.

/tikz/south right=(specification)  
Like /tikz/right but aligns the nodes at their south border. 
This is basically the same as left=of reference.south east, anchor=south west.

/tikz/west above=(specification)  
Like /tikz/above but aligns the nodes at their west border. 
This is basically the same as left=of reference.north west, anchor=south west.

/tikz/west below=(specification)  
Like /tikz/below but aligns the nodes at their west border. 
This is basically the same as left=of reference.south west, anchor=north west.

/tikz/east above=(specification)  
Like /tikz/above but aligns the nodes at their east border. 
This is basically the same as left=of reference.north east, anchor=south east.

/tikz/east below=(specification)  
Like /tikz/below but aligns the nodes at their east border. 
This is basically the same as left=of reference.south east, anchor=north east.
The same exist for the recently introduces corner anchors, too.

\[ /tikz/corner north left = \langle \text{specification} \rangle \]
- The same as \[ /tikz/north left \] but uses the new corner anchors.

\[ /tikz/corner north right = \langle \text{specification} \rangle \]
- The same as \[ /tikz/north right \] but uses the new corner anchors.

\[ /tikz/corner south left = \langle \text{specification} \rangle \]
- The same as \[ /tikz/south left \] but uses the new corner anchors.

\[ /tikz/corner south right = \langle \text{specification} \rangle \]
- The same as \[ /tikz/south right \] but uses the new corner anchors.

\[ /tikz/corner west above = \langle \text{specification} \rangle \]
- The same as \[ /tikz/west above \] but uses the new corner anchors.

\[ /tikz/corner west below = \langle \text{specification} \rangle \]
- The same as \[ /tikz/west below \] but uses the new corner anchors.

\[ /tikz/corner east above = \langle \text{specification} \rangle \]
- The same as \[ /tikz/east above \] but uses the new corner anchors.

\[ /tikz/corner east below = \langle \text{specification} \rangle \]
- The same as \[ /tikz/east below \] but uses the new corner anchors.

While the \( \langle \text{specification} \rangle \) of all these keys still accept the same form as with TiKZ, the \texttt{ext.positioning-plus} library extends this even more.

The specification after of can contain a list of coordinates (like the \texttt{fit} key of the \texttt{fit} library). This means that the new node will be placed in relation to a rectangular bounding box that fits around all this nodes in the list.

If this list is prefixed with \( | \), \(-\) or \(+\), the new node will also have the same height \( | \), the same width \(-\) or both as this bounding box.

This functionality is also available without the placement:
/tikz/fit bounding box=(list of coordinates)  (style, no default)

Creates a rectangular node with the name fit bounding box that encompasses the (list of coordinates).

/tikz/span vertical=(list of coordinates)  (style, no default)

Creates a rectangular node with the name fit bounding box that encompasses the (list of coordinates) and sets the /pgf/minimum height to the height of this bounding box.

/tikz/span horizontal=(list of coordinates)  (style, no default)

Creates a rectangular node with the name fit bounding box that encompasses the (list of coordinates) and sets the /pgf/minimum width to the width of this bounding box.

/tikz/span=(list of coordinates)  (style, no default)

Is a combination of /tikz/span vertical and /tikz/span horizontal.

As you maybe noticed in the example above, the (specification) also allows a prefix delimited by : which the node distance will be multiplied to with for the placement.  

---

5This is probably more useful when /tikz/on grid is used.
13 Scaling Pictures to a Specific Size

TikZ Library \texttt{ext.scalepicture}

\begin{verbatim}
\usetikzlibrary{ext.scalepicture} % \LaTeX and plain \TeX
\usetikzlibrary{ext.scalepicture} % Con\TeX
\end{verbatim}

This library scales TikZ pictures to a specific width or height by scaling the whole picture.

If one of the keys below are used on a TikZ picture, i.e. as an option to \tikzpicture or \begin{tikzpicture} the size of the picture\footnote{This is the size of the pseudo-node \texttt{current bounding box}.} will be measured and written to the \texttt{aux} file so that it will be available at the next compilation run and an appropriate scaling for the picture can be installed.

\begin{verbatim}
\tikzextpicturewidth
\end{verbatim}

Returns the last measured width of the picture.

This will expand to 0pt if the picture hasn’t been measured before.

\begin{verbatim}
\tikzextpictureheight
\end{verbatim}

Returns the last measured height of the picture.

This will expand to 0pt if the picture hasn’t been measured before.

\begin{verbatim}
\tikz/save picture size
\end{verbatim}

(style, no value)

This key is usually used by the keys provided by this library. Normally, this is not needed to be explicitly given.

13.1 Keeping the aspect ratio

The following \textit{unstarred} keys do not change the aspect ratio of the picture.

\begin{verbatim}
\tikz/picture width={(dimension)}  \hspace{1cm} \text{(no default)}
\end{verbatim}

Scales the picture so that the width of the picture will be (dimension). This will keep the aspect ratio the same.

\begin{verbatim}
\tikz/minimum picture width={(dimension)}  \hspace{1cm} \text{(no default)}
\end{verbatim}

As above but will not change the size of the picture if its width is less than (dimension).

\begin{verbatim}
\tikz/maximum picture width={(dimension)}  \hspace{1cm} \text{(no default)}
\end{verbatim}

As above but will not change the size of the picture if its width is greater than (dimension).

\begin{verbatim}
\tikz/picture height={(dimension)}  \hspace{1cm} \text{(no default)}
\end{verbatim}

Scales the picture so that the height of the picture will be (dimension). This will keep the aspect ratio the same.

\begin{verbatim}
\tikz/minimum picture height={(dimension)}  \hspace{1cm} \text{(no default)}
\end{verbatim}

As above but will not change the size of the picture if its height is less than (dimension).

\begin{verbatim}
\tikz/maximum picture height={(dimension)}  \hspace{1cm} \text{(no default)}
\end{verbatim}

As above but will not change the size of the picture if its height is greater than (dimension).

\begin{verbatim}
\tikz/minimum picture size={⟨width⟩}{⟨height⟩}  \hspace{1cm} \text{(no default)}
\end{verbatim}

Scales the picture so that its height will be at least (width) and its height will be at least (height).

\begin{verbatim}
\tikz/maximum picture size={⟨width⟩}{⟨height⟩}  \hspace{1cm} \text{(no default)}
\end{verbatim}

Scales the picture so that its height will be at most (width) and its height will be at most (height).

13.2 Changing the aspect ratio.

The following \textit{starred} keys do change the aspect ratio.

\begin{verbatim}
\tikz/picture width*={(dimension)}  \hspace{1cm} \text{(no default)}
\end{verbatim}

Scales the picture so that the width of the picture will be (dimension). This will only scale the $x$ axis.
/tikz/minimum picture width*=(dimension)  (no default)
As above but will not change the size of the picture if its width is greater than (dimension).

/tikz/maximum picture width*=(dimension)  (no default)
As above but will not change the size of the picture if its width is less than (dimension).

/tikz/picture height*=(dimension)  (no default)
Scales the picture so that the height of the picture will be (dimension). This will only scale the y axis.

/tikz/picture size*={(width)}{(height)}  (no default)
Scales the picture so that its width will be (width) and its height will be (height). This will scale both axes but independent from each other.
14 Arcs through Three Points

TikZ Library \texttt{ext.topaths.arcthrough}

\usetikzlibrary{ext.topaths.arcthrough} % \LaTeX{} and plain \TeX{}
\usetikzlibrary{ext.topaths.arcthrough} % \ConTeXt{}

This library allows to use an arc defined by three points.

\begin{center}
\begin{tikzpicture}
\coordinate[label=above right:$A$] (A) at ( 3, 1);
\coordinate[label=above:$B$] (B) at ( 1, 2);
\coordinate[label=below left:$C$] (C) at (-2,-2);
\draw[ultra thick, draw=green, fill=green!50] (B) to [arc through={clockwise,(A)}] (C)
\draw[ultra thick, draw=blue, fill=blue!50] (B) to [arc through=(A)] (C)
\end{tikzpicture}
\end{center}

This can only by used for circles in the canvas coordinate system.

\texttt{/tikz/arc through/through=}(coordinate) \hspace{1cm} (no default, initially (0,0))

The coordinate on the circle that defines – together with the starting and target point – a circle.

\texttt{/tikz/arc through/center suffix=}(suffix) \hspace{1cm} (no default, initially )

The arc through will define a coordinate named arc through center(suffix) so that it can be referenced later.

\texttt{/tikz/arc through/clockwise} \hspace{1cm} (no value)

The resulting arc will go clockwise from the starting point to the target point.

This will not necessarily go through the through point.

\texttt{/tikz/arc through/counter clockwise} \hspace{1cm} (no value)

The resulting arc will go counter clockwise from the starting point to the target point. This will not necessarily go through the through point.

\texttt{/tikz/arc through=}(key-value) \hspace{1cm} (no default)

This key should be used with to or edge. A parameter other than center suffix, clockwise or counter clockwise will be assumed to be the through coordinate.
15 Mirror, Mirror on the Wall

TikZ Library \texttt{ext.transformations.mirror}

\begin{verbatim}
\usetikzlibrary{ext.transformations.mirror} % \LaTeX{} and plain \TeX{}
\usetikzlibrary{ext.transformations.mirror} % Con\TeX
\end{verbatim}

This library adds more transformations to TikZ.

As explained in section 17, there are two approaches to setting a mirror transformation. As with the commands in PGF, we'll be using a lowercase \texttt{m} for the reflection matrix and an uppercase \texttt{M} for the built-in approach.

15.1 Using the reflection matrix

\begin{verbatim}
\usetikzlibrary{shapes.geometric,ext.transformations.mirror}
\begin{tikzpicture}[line join=round, thick, reg poly/.style={
  shape=regular polygon, regular polygon sides={#1}}]
\node[reg poly=5, minimum size=+2cm, draw, very thick] (a) {};
\foreach \i\[evaluate={\col=(\i-1)/.04} in {1,...,5} \]
\node \[mirror\](a.corner \i)\[mirror\](a.side \i), transform shape, reg poly=5, minimum size=+2cm, draw=red!\col!blue \] {};
\end{tikzpicture}
\end{verbatim}

\texttt{/tikz/xmirror=⟨value or coordinate⟩}

(default 0pt)

Sets up a transformation that mirrors along a horizontal line that goes through point \((⟨value⟩,0)\) or \((⟨coordinate⟩)\).
/tikz/ymirror=(value or coordinate)  
Sets up a transformation that mirrors along a vertical line that goes through point (0, ⟨value⟩) or ⟨coordinate⟩.

/tikz/mirror x=(coordinate)  
Similar to /tikz/xmirror, this however uses the xyz coordinate system instead of the canvas system.

/tikz/mirror y=(coordinate)  
Similar to /tikz/ymirror, this however uses the xyz coordinate system instead of the canvas system.

/tikz/mirror=(point A)--(point B)  
Sets up a transformation that mirrors along a line that goes through ⟨point A⟩ and ⟨point B⟩.
When only ⟨point A⟩ is given that line goes through ⟨point A⟩ and the origin.

15.2 Using built-in transformations
/tikz/xMirror=(value or coordinate)  
Sets up a transformation that mirrors along a horizontal line that goes through point ((value),0) or (coordinate).

\usetikzlibrary{ext.transformations.mirror}
\begin{tikzpicture}
\draw[help lines] (-0.25, -.25) grid (3.25, 1.25);
\draw[-latex] (0,0) .. controls (.5,1) .. (1,1);
\draw[dashed] (1.5, -.25) coordinate (m) -- (1.5, 1.25);
\draw[xMirror=(m),-latex] (0,0) .. controls (.5,1) .. (1,1);
\end{tikzpicture}

/tikz/yMirror=(value or coordinate)  
Sets up a transformation that mirrors along a vertical line that goes through point (0,(value)) or (coordinate).

/tikz/Mirror x=(coordinate)  
Similar to /tikz/xMirror, this however uses the xyz coordinate system instead of the canvas system.

\usetikzlibrary{ext.transformations.mirror}
\begin{tikzpicture}[x=.5cm, y=(45:1cm)]
\draw[-latex] (0,0) .. controls (.5,1) .. (1,1);
\draw[dashed] (1.5, -.25) coordinate (m) -- (1.5, 1.25);
\draw[xMirror=(m),-latex, red, dotted] (0,0) .. controls (.5,1) .. (1,1);
\draw[Mirror x=(m), -latex] (0,0) .. controls (.5,1) .. (1,1);
\end{tikzpicture}

/tikz/Mirror y=(coordinate)  
Similar to /tikz/yMirror, this however uses the xyz coordinate system instead of the canvas system.

/tikz/Mirror=(point A)--(point B)  
Sets up a transformation that mirrors along a line that goes through (point A) and (point B).  
When only (point A) is given that line goes through (point A) and the origin.
Part III

**PGF Libraries**

These libraries (should) work with both PGF and TikZ.
16 Arrow Tips

TikZ Library \texttt{ext.arrows}

\begin{verbatim}
\usepgflibrary{ext.arrows} % \LaTeX{} and plain \TeX{} and pure pgf
\usetikzlibrary{ext.arrows} % Con\TeX{}t and pure pgf
\usetikzlibrary{ext.arrows} % \LaTeX{} and plain \TeX{} when using TikZ
\usetikzlibrary{ext.arrows} % Con\TeX{}t when using TikZ
\end{verbatim}

This library adds arrows to \texttt{pgf}/TikZ.

Q & A: \cite{14, 8, 3, 12} & \cite{40, 28, 45, 38}

The arrow tips of the \texttt{arrows.meta} library always just touch the end of original line – which is usually what you want.

But for some arrow tips (and when they lie along a path) it makes sense that these tips shoot a bit over the end of the line. This is why these arrow tips exist. They can be categorized into three groups:

1. Centered
2. Untipped
3. Overtipped\footnote{The Overtipped arrow tips aren’t yet implemented.}

Not all original arrow tips got all variants. For a summary, refer to table on the right side. As with the original tips of the \texttt{arrows.meta} library these can be organized in the following categories.

<table>
<thead>
<tr>
<th>Group</th>
<th>Original</th>
<th>Centered</th>
<th>Untipped</th>
<th>Overtipped</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barbed</td>
<td>Arc Barb</td>
<td>\includegraphics[width=1cm]{barbed.png}</td>
<td>\includegraphics[width=1cm]{barbed.png}</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Parenthesis</td>
<td>\includegraphics[width=1cm]{parenthesis.png}</td>
<td>\includegraphics[width=1cm]{parenthesis.png}</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Hooks</td>
<td>\includegraphics[width=1cm]{hooks.png}</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Straight Barb</td>
<td>\includegraphics[width=1cm]{straight_barb.png}</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Tee Barb</td>
<td>\includegraphics[width=1cm]{tee_barb.png}</td>
<td>\includegraphics[width=1cm]{tee_barb.png}</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Bar</td>
<td>\includegraphics[width=1cm]{bar.png}</td>
<td>\includegraphics[width=1cm]{bar.png}</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Bracket</td>
<td>\includegraphics[width=1cm]{bracket.png}</td>
<td>\includegraphics[width=1cm]{bracket.png}</td>
<td>\includegraphics[width=1cm]{bracket.png}</td>
</tr>
<tr>
<td>Geometric</td>
<td>Circle</td>
<td>\includegraphics[width=1cm]{circle.png}</td>
<td>\includegraphics[width=1cm]{circle.png}</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Ellipse</td>
<td>\includegraphics[width=1cm]{ellipse.png}</td>
<td>\includegraphics[width=1cm]{ellipse.png}</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Kite</td>
<td>\includegraphics[width=1cm]{kite.png}</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Diamond</td>
<td>\includegraphics[width=1cm]{diamond.png}</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Turned Square</td>
<td>\includegraphics[width=1cm]{turned_square.png}</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>LaTeX</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Square</td>
<td>\includegraphics[width=1cm]{square.png}</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Rectangle</td>
<td>\includegraphics[width=1cm]{rectangle.png}</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Stealth</td>
<td>\includegraphics[width=1cm]{stealth.png}</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Triangle</td>
<td>\includegraphics[width=1cm]{triangle.png}</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Rays</td>
<td>\includegraphics[width=1cm]{rays.png}</td>
<td>\includegraphics[width=1cm]{rays.png}</td>
<td>–</td>
</tr>
</tbody>
</table>
16.1 Centered

16.1.1 Barbed Arrow Tips

**Arrow Tip Kind Centered Arc Barb**
This is a variant of the Arc Barb tip. The center of the arc lies on the original end of the path.

**Arrow Tip Kind Centered Bar**
A variant of the simple Bar tip. This is a simple instance of Centered Tee Barb for length zero. The middle of the line will lie on original end of the path.

**Arrow Tip Kind Centered Bracket**
This is a variant of the Bracket tip and therefore an instance of the Centered Tee Barb arrow tip that results in something resembling a bracket. The middle of the vertical part will lie on the original end of the path.

**Arrow Tip Kind Centered Hooks**
A variant of the Hooks tip. The starting point of the hooks will lie on the original end of the path.

**Arrow Tip Kind Centered Parenthesis**
This is a variant of the Parenthesis tip and thus an instance of the Centered Arc Barb arrow tip.

**Arrow Tip Kind Centered Straight Barb**
A variant of the Straight Barb tip.

**Arrow Tip Kind Centered Tee Barb**
A variant of the Tee Barb tip. The middle of the vertical part will lie on the original end of the path.

16.1.2 Geometric Arrow Tips

**Arrow Tip Kind Centered Circle**
A variant of the Circle tip. The center of the circle will lie on the original end of the path.

**Arrow Tip Kind Centered Diamond**
This is a variant of the Diamond tip and thus an instance of Centered Kite where the length is larger than the width.

**Arrow Tip Kind Centered Ellipse**
This is a variant of the Ellipse tip and thus another name for the Centered Circle tip that is twice as wide as high.

**Arrow Tip Kind Centered Kite**
A variant of the Kite tip. The widest part will lie on the original end of the path.

**Arrow Tip Kind Centered Rectangle**
A variant of the Rectangle tip. By default, it is twice as long as high.

**Arrow Tip Kind Centered Square**
A variant of the Square tip.

**Arrow Tip Kind Centered Stealth**
This is a variant of the Stealth tip. The weighted center will lie at the original end of the path.

**Arrow Tip Kind Centered Triangle**
This is a variant of the Triangle tip and thus an instance of the Centered Kite tip with zero inset.

**Arrow Tip Kind Centered Turned Square**
This is a variant of the Turned Square tip and thus an instance of the Centered Kite tip with identical width and height and mid-inset.

16.1.3 Special Arrow Tips

**Arrow Tip Kind Centered Rays**
A variant of the Rays tip. The origin of the rays will lie on the original end of the path.
16.2 Untipped

16.2.1 Barbed Arrow Tips

**Arrow Tip Kind** Centered Arc Barb
This is a variant of the Arc Barb tip. The arrow tip will protrude half its line width over the original end of the path.

**Arrow Tip Kind** Untipped Bar
A variant of the simple Bar tip. This is a simple instance of Untipped Tee Barb for length zero.
The middle of the line will lie on original end of the path.

**Arrow Tip Kind** Untipped Bracket
This is a variant of the Bracket tip and therefore an instance of the Untipped Tee Barb arrow tip that results in something resembling a bracket.
The arrow tip will protrude half its line width over the original end of the path.

**Arrow Tip Kind** Untipped Parenthesis
This is a variant of the Parenthesis tip and thus an instance of the Untipped Arc Barb arrow tip.

**Arrow Tip Kind** Untipped Tee Barb
A variant of the Tee Barb tip.
The middle of the vertical part will lie on the original end of the path.

16.2.2 Geometric Arrow Tips

**Arrow Tip Kind** Untipped Circle
A variant of the Circle tip. This tip will protrude half its line width over the original end of the path.

**Arrow Tip Kind** Untipped Ellipse
This is a variant of the Ellipse tip and thus another name for the Untipped Circle tip that is twice as wide as high.

16.3 Original Arrow Tips

**Arrow Tip Kind** Hug Cap
This arrow tips will hug a circle that would touch the end of the path.
Use the /pgf/arrow keys/length key to set up the radius of that circle.

**Arrow Tip Kind** Loop
This arrow tip attaches a one-sided loop to the end of the line. The length refers to the length of the whole tip while the radius specifies the radius of the three rounded corners. The width of the tip is twice the length (but can’t specified independently).

<table>
<thead>
<tr>
<th>Appearance of the below at line width</th>
<th>0.4pt</th>
<th>0.8pt</th>
<th>1.6pt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loop[</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Loop[sep] Loop[</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The following options have no effect: harpoon, round, line width. On double lines, the arrow tip will not look correct.
17 Transformations: Mirroring

PGF Library ext.transformations.mirror

\usepgflibrary{ext.transformations.mirror} % \LaTeX and \texttt{plain TeX}
\usepgflibrary{ext.transformations.mirror} % ConTeXt

This library adds mirror transformations to PGF.

Two approaches to mirror transformation exist:

1. Using the reflection matrix (see left column).
   This depends on \texttt{\pgfpointnormalised} which involves the sine and the cosine functions of PGF\texttt{math}.

2. Using built-in transformations (see right column).
   This depends on \texttt{\pgfmathanglebetweenpoints} which involves the arctangent (\texttt{atan2}) function of PGF\texttt{math}.

Which one is better? I don’t know. Choose one you’re comfortable with.

17.1 Using the reflection matrix

The following commands use the reflection matrix that sets the transformation matrix following

\[
A = \frac{1}{|\vec{l}|^2} \begin{bmatrix}
  l_y^2 - l_x^2 & 2l_x l_y \\
  2l_x l_y & l_x^2 - l_y^2
\end{bmatrix}.
\]

\texttt{\pgftransformxmirror{⟨value⟩}}

Sets up a transformation that mirrors along a vertical line that goes through point \((⟨value⟩, 0)\).

\begin{tikzpicture}
\draw[help lines] (-.25, -.25) grid (3.25, 1.25);
\draw[-latex] (0,0) .. controls (.5,1) .. (1,1);
\draw[dashed] (1.5, -.25) -- (1.5, 1.25);
\pgftransformxmirror{1.5}
\draw[-latex] (0,0) .. controls (.5,1) .. (1,1);
\end{tikzpicture}

17.2 Using built-in transformations

The following commands use a combination of shifting, rotating, \(-1\) scaling, rotating back and shifting back to reach the mirror transformation.

The commands are named the same as on the left side, only the \texttt{m} in \texttt{mirror} is capitalized.

\texttt{\pgftransformxMirror{⟨value⟩}}

Sets up a transformation that mirrors along a vertical line that goes through point \((⟨value⟩, 0)\).

\begin{tikzpicture}
\draw[help lines] (-.25, -.25) grid (3.25, 1.25);
\draw[-latex] (0,0) .. controls (.5,1) .. (1,1);
\draw[dashed] (1.5, -.25) -- (1.5, 1.25);
\pgftransformxMirror{1.5}
\draw[-latex] (0,0) .. controls (.5,1) .. (1,1);
\end{tikzpicture}
\texttt{\textbackslash pgftransformymirror\{\textit{value}\}}

Sets up a transformation that mirrors along a horizontal line that goes through point \((0, \langle \textit{value} \rangle)\).

\texttt{\textbackslash pgftransformmirror\{\langle \textit{point A} \rangle\}{\langle \textit{point B} \rangle}}

Sets up a transformation that mirrors along the line that goes through \langle \textit{point A} \rangle and \langle \textit{point B} \rangle.

\texttt{\textbackslash usepgflibrary\{transformations.mirror\}}
\begin{tikzpicture}
\draw[help lines] (-0.25, -2.25) grid (2.5, 1.25);
\draw[latex-] (0,0) .. controls (.5,1) .. (1,1);
\draw[dashed] (0, -1) -- (2, 0);
\pgftransformmirror{\pgfpointxy{0}{-1}}{\pgfpointxy{2}{0}}
\draw[latex-] (0,0) .. controls (.5,1) .. (1,1);
\end{tikzpicture}

\texttt{\textbackslash pgfqtransformmirror\{\langle \textit{point A} \rangle\}}

Sets up a transformation that mirrors along the line that goes through the origin and \langle \textit{point A} \rangle.

\texttt{\textbackslash usepgflibrary\{transformations.mirror\}}
\begin{tikzpicture}
\draw[help lines] (-0.25, -2.25) grid (2.5, 1.25);
\draw[latex-] (0,0) .. controls (.5,1) .. (1,1);
\draw[dashed] (0, -1) -- (2, 0);
\pgfqtransformmirror{\pgfpointxy{0}{-1}}{\pgfpointxy{2}{0}}
\draw[latex-] (0,0) .. controls (.5,1) .. (1,1);
\end{tikzpicture}
18 Shape: Circle Arrow

TikZ Library **ext.shapes.circlearrow**

\usepgflibrary{ext.shapes.circlearrow} % \LaTeX{} and plain \TeX{} and pure pgf
\usepgflibrary[ext.shapes.circlearrow] % Con\TeX{}t and pure pgf
\usetikzlibrary{ext.shapes.circlearrow} % \LaTeX{} and plain \TeX{} when using TikZ
\usetikzlibrary[ext.shapes.circlearrow] % Con\TeX{}t when using TikZ

A circular shape named **circle arrow** that has an arc as its background path that can have an arrow tip.

**Q & A:** [24] & [42]

**Shape circle arrow**

This shape is an arrow whose path is an arc – defined very similar to the arc path operation – that can possibly be customized with arrow tips.

\[
\text{/pgf/circle arrow start angle} = \text{(start angle)}
\]
Sets the start angle.

\[
\text{/pgf/circle arrow end angle} = \text{(end angle)}
\]
Sets the end angle.

\[
\text{/pgf/circle arrow delta angle} = \text{(delta angle)}
\]
Sets the delta angle.

\[
\text{/pgf/circle arrow arrows} = \text{(start arrow tip specification)} - \text{(end arrow tip specification)}
\]
The specification will be forwarded to \pgfsetarrows .

A few handful styles are pre-defined.

\[
\text{/pgf/circle arrow turn left north}
\]
Sets circle arrow start angle = 100, circle arrow delta angle = 340 and circle arrow arrows = -. (no default, initially )

\[
\text{/pgf/circle arrow turn left east}
\]
As above but circle arrow start angle = 10. (no value)

\[
\text{/pgf/circle arrow turn left west}
\]
As above but circle arrow start angle = 280. (no value)

\[
\text{/pgf/circle arrow turn left south}
\]
As above but circle arrow start angle = 190. (no value)
/pgf/circle arrow turn right north
Sets circle arrow start angle = 100, circle arrow delta angle = 340 and circle arrow arrows = <-. (no value)

/pgf/circle arrow turn right east
As above but circle arrow start angle = 10. (no value)

/pgf/circle arrow turn right west
As above but circle arrow start angle = 280. (no value)

/pgf/circle arrow turn right south
As above but circle arrow start angle = 190. (no value)

\usetikzlibrary{ext.shapes.circlearrow,matrix}
\begin{tikzpicture}
\matrix[matrix of nodes, draw=none, row sep=1em, column sep=1em,
every node/.style={draw=gray, shape=circle arrow, ultra thick, inner sep=1em}]
(m) { [circle arrow turn left north] & [circle arrow turn left east] \\ [circle arrow turn left west] & [circle arrow turn left south] \\ [circle arrow turn right north] & [circle arrow turn right east] \\ [circle arrow turn right west] & [circle arrow turn right south] \\ }; \end{tikzpicture}
\usetikzlibrary{ext.shapes.circlearrow}
\begin{tikzpicture}
\node[name=s, shape=\texttt{circle arrow},
circle arrow turn left west, shape example]
{Circle Arrow\vrule width 1pt height 2cm};
\foreach \anchor/\placement in
{north west/above left, north/above, north east/above right, west/left, center/above, east/right, mid west/right, mid/above, mid east/left, base west/left, base/below, base east/right, south west/below left, south/below, south east/below right, text/left, 10/right, 130/above}
\draw[shift=(s.\anchor)] plot[mark=x] coordinates{(0,0)} node[\placement] {\scriptsize\texttt{(s.\anchor)}};
\end{tikzpicture}
19 Shape: Circle Cross Split

TikZ Library `ext.shapes.circlecrosssplit`

The `circle cross split` shape is a circular shape with four parts that can be individually filled.

**Q&A:** [18] & [43]

**Shape circle cross split**

This shape has four node parts that are placed near the center of a circle.

- `/pgf/circle cross split part fill={⟨list⟩}`
  Sets the custom fill color for each node part shape. The items in ⟨list⟩ should be separated by commas (so if there is more than one item in ⟨list⟩, it must be surrounded by braces). If ⟨list⟩ has less entries than node parts, then the remaining node parts use the color from the last entry in the list. This key will automatically set `/pgf/circle cross split uses custom fill`.

- `/pgf/circle cross split uses custom fill=⟨boolean⟩` (default true)
  This enables the use of a custom fill for each of the node parts (including the area covered by the `inner sep`). The background path for the shape should not be filled (e.g., in TikZ, the fill option for the node must be implicitly or explicitly set to none). Internally, this key sets the TeX-if\ifpgfcirclecrosssplitcustomfill appropriately.
text two
three four
20 Shape: Heatmark

TikZ Library `ext.shapes.heatmark`

\usepgflibrary{ext.shapes.heatmark} % \LaTeX{} and plain \TeX{} and pure pgf
\usepgflibrary[ext.shapes.heatmark] % Con\TeX{} and pure pgf
\usetikzlibrary{ext.shapes.heatmark} % \LaTeX{} and plain \TeX{} when using Ti\kZ
\usetikzlibrary[ext.shapes.heatmark] % Con\TeX{} when using Ti\kZ

A circular shape that has customizable rings around it.

Q & A: [2] & [33]

Shape heatmark

\begin{itemize}
  \item `/pgf/heatmark arcs=(arcs num)` (no default, initially 3)
    \begin{itemize}
      \item Sets the number of arc around the circle to `(arcs num)`.\end{itemize}
  \item `/pgf/heatmark arc width=(arc width)` (no default, initially 4pt)
    \begin{itemize}
      \item Sets the width of the rings around the circle to `(arc width)`.\end{itemize}
  \item `/pgf/heatmark arc sep=(sep length)` (no default, initially 1pt)
    \begin{itemize}
      \item Sets the whitespace between the rings to `(sep length)`.\end{itemize}
  \item `/pgf/heatmark arc rings=(rings num)` (no default, initially 3)
    \begin{itemize}
      \item Sets the number of rings around the circle to `(rings num)`.\end{itemize}
  \item `/pgf/heatmark arc sep angle=(sep angle)` (no default, initially 20)
    \begin{itemize}
      \item Sets the whitespace angle between the arcs in one ring to `(sep angle)`.\end{itemize}
  \item `/pgf/heatmark inner opacity=(inner opacity)` (no default, initially 0.8)
    \begin{itemize}
      \item Sets the opacity of the inner ring to `(inner opacity)`.\end{itemize}
  \item `/pgf/heatmark outer opacity=(low opacity)` (no default, initially 0.2)
    \begin{itemize}
      \item Sets the opacity of the outer ring to `(outer opacity)`.\end{itemize}
      \item The opacity of the rings between the outer and the inner ring will be interpolated by these two opacities.\end{itemize}
\end{itemize}

This shape takes the value of `/pgf/shape border rotate` into consideration.

For every ring and for every arc the following style keys are tried.

`/pgf/heatmark ring (ring number)` (style, no value)
The \texttt{pgf} shape is set up in a way that even TikZ styles can be used with a little bit of work:

```latex
\usetikzlibrary{ext.shapes.heatmark}
\tikz[
    shape border rotate=90,
    /pgf/heatmark ring 1/.append style={/tikz/fill=green},
    /pgf/heatmark arc 1/.append style={/tikz/fill=blue},
    /pgf/heatmark ring 2 arc 2/.append style={/tikz/fill=yellow!70!black}
] 
ode[heatmark, fill=red] (n) {100};
```

It is best to use this shape with no actual border (\texttt{draw = none}) and the \texttt{outer sep} set to zero.
21 Shape: Rectangle with Rounded Corners

TikZ Library `ext.shapes.rectangleroundedcorners`

\usepgflibrary{ext.shapes.rectangleroundedcorners} % \LaTeX{} and plain \TeX{} and pure pgf
\usetikzlibrary{ext.shapes.rectangleroundedcorners} % \LaTeX{} and plain \TeX{} when using TikZ
\usetikzlibrary[ext.shapes.rectangleroundedcorners] % Con\TeXt{} when using TikZ

A rectangle with rounded corners.

Shape rectangle with rounded corners

This library provides a rectangle with rounded corners where every corner can have a different radius.

\begin{itemize}
  \item /pgf/rectangle with rounded corners north west radius=⟨\texttt{dimen}⟩
    \hspace{1em} Sets the north west radius to ⟨\texttt{dimen}⟩.
  \item /pgf/rectangle with rounded corners north east radius=⟨\texttt{dimen}⟩
    \hspace{1em} Sets the north east radius to ⟨\texttt{dimen}⟩.
  \item /pgf/rectangle with rounded corners south west radius=⟨\texttt{dimen}⟩
    \hspace{1em} Sets the south west radius to ⟨\texttt{dimen}⟩.
  \item /pgf/rectangle with rounded corners south east radius=⟨\texttt{dimen}⟩
    \hspace{1em} Sets the south east radius to ⟨\texttt{dimen}⟩.
  \item /pgf/rectangle with rounded corners radius=⟨\texttt{dimen}⟩
    \hspace{1em} Sets all radii to ⟨\texttt{dimen}⟩.
\end{itemize}

(no default, initially .5\pgflinewidth)
\usepgflibrary{ext.shapes.rectangleroundedcorners}
\begin{tikzpicture}
\Huge
\node[name=s, shape=rectangle with rounded corners, shape example, rectangle with rounded corners north west radius=10pt, rectangle with rounded corners north east radius=20pt, rectangle with rounded corners south west radius=30pt, rectangle with rounded corners south east radius=40pt] {Rectangle with rounded corners\vrule width 1pt height 2cm};
\foreach \anchor in 
{north west/above left, north/above, north east/above right, west/left, center/above, east/right, mid west/right, mid/above, mid east/left, base west/left, base/below, base east/right, south west/below left, south/below, south east/below right, text/below, 10/right, 130/above, north west center/below right, north east center/left, south west center/above right, south east center/left, below north west/left, above south west/left, above south east/right, below north east/right, right north west/above, right south west/below, left south east/below, left north east/above}
\draw[shift={s.\anchor}] plot[mark=x] coordinates{(0,0)} node[placement] {\scriptsize\texttt{(s.\anchor)}};
\end{tikzpicture}
22 Shape: Superellipse

TikZ Library `ext.shapes.superellipse`

\usepgflibrary{ext.shapes.superellipse} % \LaTeX{} and plain \TeX{} and pure pgf
\usetikzlibrary{ext.shapes.superellipse} % \LaTeX{} and plain \TeX{} when using TikZ
\usetikzlibrary{ext.shapes.superellipse} % ConTExt when using TikZ

Shape in the form of a “superellipse”.

Q & A: [49] & [27]

Shape superellipse

This shape is defined by formula

\[ |\frac{x}{r_x}|^m + |\frac{y}{r_y}|^n = 1 \]

and will be plotted by

\[ x(t) = |\cos t|^{\frac{2}{m}} \cdot r_x \sgn(\cos t) \]
\[ y(t) = |\sin t|^{\frac{2}{n}} \cdot r_y \sgn(\sin t) \]

where \( r_x \) is half the node’s width and \( r_y \) is half the node’s height.

/\texttt{pgf/superellipse \texttt{x exponent}}=(\texttt{x exponent}) \hfill (no default, initially 2.5)

This sets \( m \).

/\texttt{pgf/superellipse \texttt{y exponent}}=(\texttt{y exponent}) \hfill (no default, initially 2.5)

This sets \( n \).

/\texttt{pgf/superellipse \texttt{step}}=(\texttt{step}) \hfill (no default, initially 5)

This specifies the step of the underlying plot handler. The smaller (\texttt{step}) is, the slower computation will be.

Sensible values for (\texttt{step}) are integer dividers of 90, i.e. 2, 3, 5, 6, 9, 10, 15, 18, 30 and 45.

/\texttt{pgf/superellipse \texttt{exponent}}=(\texttt{exponent}) \hfill (no default)

Sets both superellipse \texttt{x exponent} and superellipse \texttt{y exponent} to (\texttt{exponent}).
Notes on Implementation  For implementing this shape, additional mathematical functions were declared.

\text{superellipsex}(t, \frac{2}{m}, r_x)\newline
\text{"pgfmathsuperellipsex}{t}{\frac{2}{m}}{r_x}\newline
Returns the \(x\) value on a point of the superellipse with its center on the origin following

\[ x = r_x \cos^{2/m} t \]

for values of \(0 \leq t \leq 90\).

\text{superellipsey}(t, \frac{2}{n}, r_y)\newline
\text{"pgfmathsuperellipsey}{t}{\frac{2}{n}}{r_y}\newline
Returns the \(y\) value on a point of the superellipse with its center on the origin following

\[ y = r_y \cos^{2/n} t \]

for values of \(0 \leq t \leq 90\).

Both \texttt{pgfmath} functions can be used at once with the following macro.

\text{"pgfmathsuperellipseXY}{\langle t \rangle}{\langle \frac{2}{m} \rangle}{\langle \frac{2}{n} \rangle}{\langle a \rangle}{\langle b \rangle}\newline
Returns the \(x\) value (in \texttt{\pgfmathresultX}) and the \(y\) value (in \texttt{\pgfmathresultY}) of the superellipse with its center on the origin following

\[ x = a \cos^{2/m} t \\
\[ y = b \cos^{2/n} t \]

for values of \(0 \leq t \leq 90\).

Note: all arguments must be a valid number since they will not be parsed by \texttt{pgfmath}.

And additional internal macro was defined following the original naming scheme.

\text{"pgfutil@prefix@macrotomacro}{\langle macro 1 \rangle}{\langle macro 2 \rangle}\newline
Adds the once-expansion of \( \langle macro 2 \rangle \) in front of \( \langle macro 1 \rangle \).
\usetikzlibrary{ext.shapes.superellipse}
\begin{tikzpicture}
\node[name=s,shape=superellipse,shape example] {
  \Huge Superellipse
  \vrule width 1pt height 2cm};
\foreach \anchor/\placement in 
  {north west/above left, north/above, north east/above right, 
    west/left, center/above, east/right, 
    mid west/right, mid/above, mid east/left, 
    base west/left, base/below, base east/right, 
    south west/below left, south/below, south east/below right, 
    text/left, 10/right, 130/above}
{\draw[shift={s.\anchor}] plot[mark=x] coordinates{(0,0)}
  node[\placement] \scriptsize{(s.\anchor)};}
\end{tikzpicture}
\usetikzlibrary{ext.shapes.superellipse}
\begin{tikzpicture}
\foreach \xe/\ye[count=i] in {0.5/0.5, 1/1, 2/2, 3/3, 0.5/5}
{\node[draw, superellipse, superellipse x exponent=\xe, superellipse y exponent=\ye] at (1.5*i,0) {}}
\end{tikzpicture}
23 Shape: Uncentered Rectangle

PGF Library ext.shapes.uncenteredrectangle

\usepgflibrary{ext.shapes.uncenteredrectangle} \% \LaTeX{} and plain \TeX
\usepgflibrary[ext.shapes.uncenteredrectangle] \% Con\TeX

A rectangle that has a variable horizontal center with three node parts.

Q & A: [52, 23] & [34, 31]

Shape uncentered rectangle

For some alignment problems, this shape could be useful.

It has three node parts: the standard text part, the left part that is to the left of text and the right part that is to the right of text.

When edges are to be connected with this shape, the following key changes to which inner center this shape will calculate the appropriate point on the border.

\texttt{/pgf/uncentered rectangle center\{left\} or \{text\} or \{right\} or \{real\}} \hfill (no default, initially text)

Sets the center that is to be used for connecting edges.

This will also move the anchors north, mid, base and south along. In the picture below, this are marked red.

\texttt{/pgf/uncentered rectangle use saved center\{true\} or \{false\}} \hfill (default true)

When this is set to true, the border anchors will use the horizontal center that was used when the node was created.

For support of the cd library of the \texttt{tikz-cd} package, this shape also supports a dynamic \texttt{y} value for its anchors center, west and east.

\texttt{/pgf/uncentered rectangle center yshift\{dimension\}} \hfill (no default, initially {})

This determines the distance between the baseline and the center anchors.

If \texttt{(dimension)} is empty, the real vertical center will be used.

For use with \texttt{cd}, set this to \texttt{axis_height}.
TikZ Library `ext.shapes.uncenteredrectangle`

\usetikzlibrary{ext.shapes.uncenteredrectangle} \% \LaTeX{} and plain \TeX{}
\usetikzlibrary{ext.shapes.uncenteredrectangle} \% Con\TeX{}

This library extends the cd library (from the tikz-cd package) so that it can be used with the uncentered rectangle shape.

Q: [50]

This library provides only one key.

/tikz-ext/tikz-cd fix

This key installs various “fixes” to the /tikz/commutative diagrams/every diagram style:

- Firstly, is defines a /tikz/matrix of math nodes key (only for the tikzcd environment) which allows to toggle the /tikz/commutative diagrams/math mode for...
each node.\footnote{Due to a bug with /tikz/execute at end node, the "automatic" math mode in matrices can't be used with multipart nodes.}

- The helpful macro `\uncrec` will be installed.

\begin{verbatim}
\uncrec{(left)}{(center)}{(right)}
\end{verbatim}

When used as the content of uncentered rectangle, the node parts will be setup so that \texttt{(left)} is in the left part of the node part etc.

- Since math mode will be disabled with the uncentered rectangle, it is automatically enabled for each node part with `\uncrec` but it can be disabled with the following key.

\begin{verbatim}
/tikz/uncrec math mode=(true) or (false)
\end{verbatim}

\begin{verbatim}
When enabled the contents of `\uncrec` will be set in math mode.
\end{verbatim}

- For easy access to the uncentered rectangle shape, the following keys are available inside a Commutative Diagram.

\begin{verbatim}
/tikz/uncrec=(left) or (text) or (right) or (real)
\end{verbatim}

\begin{verbatim}
This key sets the shape to uncentered rectangle and /pgf/uncentered rectangle center to its argument.
\end{verbatim}

\begin{verbatim}
/tikz/commutative diagrams/install uncentered rectangle in columns=(column)
\end{verbatim}

\begin{verbatim}
All nodes in column \texttt{(column)} will be set to the uncentered rectangle shape.
\end{verbatim}
Part IV
Utilities

\usetikzlibrary{ext.misc}
\begin{tikzpicture}
\declarefunction={bigR(n)=smallR+.05*n;}
\declareconstant={smallR=1; segments=20;},
\full arc=segments
\foreach \iN[evaluate={\endRadius=bigR(\iN+1);}, use int=0 to segments-1]
\filldraw[fill=gray!50] (\iN R:\endRadius)
\arc[radius=\endRadius, start angle=\iN R, delta angle=1R] -- (\iN R+1R:smallR)
\arc[radius=smallR, end angle=\iN R, delta angle=-1R] -- cycle;
\node {$\phi^2$};
\node at (north west:{sqrt 2 * bigR(segments/2)}) {$\{\Omega\}_{i=1}^n$};
\node[rotate=-.5R, right] at (-.5R: bigR segments) {$\partial \varphi$};
\tikzset{yshift=-5cm, declare constant={segments=25;}, full arc=segments}
\filldraw[fill=gray!50] (right:smallR)
\foreach \iN[evaluate={\endRadius=bigR(\iN+1);}, use int=0 to segments-1] \arc[radius=\endRadius, start angle=\iN R, delta angle=1R]
\arc[radius=smallR, start angle=0, delta angle=-360];
\node {$\phi^2$};
\node at (north west:{sqrt 2 * bigR(segments/2)}) {$\{\Omega\}_{i=1}^n$};
\node[rotate=-.5R, right] at (-.5R: bigR segments) {$\partial \varphi$};
\end{tikzpicture}
24 Calendar: Weeknumbers and more conditionals

\usepackage{calendar-ext} % \LaTeX
\input calendar-ext.tex % plain \LaTeX

This package adds week numbers and more conditionals to the PGF package pgfcalendar.

Q & A: [10, 11, 15] & [26, 46, 32]

24.1 Extensions

The following tests are added.

- **Jan** This test is passed by all dates that are in the month of January.
- **Feb** as above.
- **Mar** as above.
- **Apr** as above.
- **May** as above.
- **Jun** as above.
- **Jul** as above.
- **Aug** as above.
- **Sep** as above.
- **Oct** as above.
- **Nov** as above.
- **Dec** as above.

- **leap year=⟨year⟩** This test checks whether the given year is a leap year. If ⟨year⟩ is omitted, it checks the year of the current date.

- **and=⟨tests⟩** This test passes when all ⟨tests⟩ pass.

- **not=⟨tests⟩** This test passes when ⟨tests⟩ do not pass.

- **week of month=⟨num⟩** This test passes when the date is in ⟨num⟩th week of the month. The first week of the month start at day 1 and ends with day 7.

- **week of month'=⟨num⟩** As above but counts from the last day of the month. For a month with 31 days, this means the “1st” week starts at day 25 and ends with day 31.

- **calendar week of month=⟨num⟩** This test passes when the date is in ⟨num⟩th calendar week of the month. The first week starts at the first day of the month and ends at the next Sunday.

- **calendar week of month'=⟨num⟩** As above but counts from the last day of the month.

- **yesterday=⟨tests⟩** This test passes when the previous day passes ⟨tests⟩.

- **week=⟨num⟩** This test passes when the current week of the year equals ⟨⟨num⟩⟩.

The shorthands for d- and m- are slightly changed so that they are expandable. This makes it possible to use these shorthands inside of PGFmath. The shorthands for the week (see section 24.2) are added. These are

- **n-** (shortest numerical representation),
- **n=** (shortest but added horizontal space) and
- **n0** (leading zero when below 10).
24.2 Week numbering (ISO 8601)

\texttt{\textbackslash pgfcalendarjulianyeartoweek{\textbackslash Julian day}{\textbackslash year}{\textbackslash week count}}

This command calculates the week for the \textit{Julian day} of \textit{year}. The \textit{week counter} must be a \LaTeX{} count.

The calculation follows the rule of ISO 8601 where the first week has that year’s first Thursday in it.

Inside of \texttt{\textbackslash pgfcalendar} the command \texttt{\textbackslash pgfcalendarcurrentweek} will be available.

\texttt{\textbackslash pgfcalendarcurrentweek}

This command returns the current week number (always two digits – use shorthand \texttt{n} to strip the leading zero).

Inside of \texttt{\textbackslash ifdate} the command \texttt{\textbackslash pgfcalendarifdateweek} will be available.

\texttt{\textbackslash pgfcalendarifdateweek}

This command returns the week number (always two digits).
This package adds small niceties to the `pgffor` package. Most of these additions are also available with the `ext.misc` library.

**Warning:** Consider this package experimental. At the very least, it will break the \ldots notation and possibly gobbles spaces after the body.

Q & A: [1, 7, 53] & [35, 41, 37]

Instead of `\foreach \var in {start, start + delta, \ldots, end}` one can use `\foreach \var[use int=start to end step delta]`.

\[ /pgf/foreach/use_int=(start)to(end)step(delta) \]  
(no default)

The values `(start), (end) and (delta)` are evaluates by `PGFmath` at initialization. The part `step (delta)` is optional (`(delta) = 1`).

\[ /pgf/foreach/use_float=(start)to(end)step(delta) \]  
(no default)

Same as above, however the results are not truncated.

\[ /pgf/foreach/no separator \]  
(no value)

This key disables any separator between elements of the list. Every token is its own element. This also means that Unicode characters need to be grouped between { and } if Lua\TeX{} isn’t used. Spaces will be ignored.

\[ /pgf/foreach/normal list \]  
(no value)

This key simply disables all other special parsers and returns to the original list parser.
The following keys only work with \LaTeX{} and cannot be used when only the ext.misc library or the plain\TeX{} pgffor-ext.tex are loaded. For this, you will need to use \texttt{\usepackage{pgffor-ext}}.

\texttt{/pgf/foreach/xparser=}\{\langle argument specification\rangle\}\{\langle foreach value\rangle\}

This key can be used to specify a \texttt{xparse} specification for each element in the list.

For this to work somewhat seamless, the following needs to observed:

- Every \{\langle argument specification\rangle\} get appended \texttt{u,}. This means there’s always one additional mandatory argument at the end of every element.
- The \{\langle foreach value\rangle\} needs to correspond to the /pgf/foreach/var value.

\texttt{/pgf/foreach/xparser 0m}

Sets up a list whose elements may contain an optional argument inside \[] which correspond to two \texttt{\foreach} variables, say \texttt{\Options/\Text}.

\textbf{Key handler} \texttt{(key)/.list xparse=}\{\langle argument specification\rangle\}\{\langle comma-separated list of values\rangle\}

This handler causes the key to be used repeatedly, namely once for every element of the list of values. The \{\texttt{comma-separated list of values}\} is processed using \texttt{\foreach} and the given \texttt{xparse} \{\texttt{argument specification}\} with the aforementioned \texttt{xparser} key.
26 And a little bit more

TikZ Library \texttt{ext.miss}

\usetikzlibrary{ext.miss} % \LaTEX and plain \TeX
\usetikzlibrary{ext.miss} % Con\TeX

This library adds miscellaneous utilities to \texttt{pgfmath}, \texttt{PGF} or \texttt{TikZ}.

Q & A: [22] & [25]

26.1 \texttt{PGFmath}

26.1.1 Postfix operator \texttt{R}

Similar to \texttt{\segments[<num>]} in \texttt{PSTricks}, the postfix operator \texttt{R} allows the user to use an arbitrary number of segments of a circle to be used instead of an angle.

\texttt{/pgf/full arc=(num)} \hspace{1cm} \texttt{(default \{}{}\texttt{)}

The number \texttt{(num)} of segments will be set up. Using \texttt{full arc} with an empty value disables the segmentation and \texttt{1R} equals \(1^\circ\).

The given value \texttt{(num)} is evaluated when the key is used and doesn’t change when \texttt{(num)} contains variables that change.

The \texttt{R} operator can then be used.

\texttt{xR} \hspace{1cm} \texttt{(postfix operator; uses the \texttt{fullarc} function)}

Multiplies \texttt{x} with \(\frac{360}{\texttt{(num)}}\).

26.1.2 Functions

\texttt{strrepeat("Text", x)}

\texttt{\pgfmathstrrepeat{"Text"}{x}}

Returns a string with \texttt{Text} repeated \texttt{x} times.

\texttt{foofoofoofoofoo} \hspace{1cm} \texttt{\pgfmathparse{strrepeat("foo", 5)}} \hspace{1cm} \texttt{\pgfmathresult}

\texttt{isInString("String", "Text")}

\texttt{\pgfmathisInStrin{"String"}{"Text"}}

Returns \texttt{1} (true) if \texttt{Text} contains \texttt{String}, otherwise \texttt{0} (false).

\texttt{strcat("Text A", "Text B", \ldots)}

\texttt{\pgfmathstrcat{"Text A"}{"Text B"}{\ldots}}

Returns the concatenation of all given parameters.

\texttt{\texttt{blue!21!green}} \hspace{1cm} \texttt{\pgfmathparse{strcat("blue!", int(7*3), "!green")}} \hspace{1cm} \texttt{\pgfmathresult}

\texttt{isEmpty("Text")}

\texttt{\pgfmathisEmpty{"Text"}}

Returns \texttt{1} (true) if \texttt{Text} is empty, otherwise \texttt{0} (false).

\texttt{\texttt{0 and 1}} \hspace{1cm} \texttt{\pgfmathparse{isEmpty("foo") \pgfmathresult \ and\ \pgfmathparse{isEmpty("bar")} \pgfmathresult}} \hspace{1cm} \texttt{\pgfmathresult}

\texttt{atanXY(x,y)}
Arctangent of $y \div x$ in degrees. This also takes into account the quadrant. This is just a argument-swapped version of `atan2` which makes it easier to use the \p commands of the `calc` library.

\pgfmathatanXY{x}{y}

atanXY(y,x)

Arctangent of $y \div x$ in degrees. This also takes into account the quadrant.

\pgfmathatanXY{y}{x}

26.1.3 Functions: using coordinates

The following functions can only be used with PGF and/or TikZ. Since the arguments are usually plain text (and not numbers) one has to wrap them in ".

\usetikzlibrary{calc,ext.misc,through}
\begin{tikzpicture}
\path (0,0) coordinate (A) + (0:4) coordinate (B) +(75:4) coordinate (C);
draw (A) -- (B) -- (C) -- cycle;
\foreach \cnt in {1,...,4} {
  \pgfmathsetmacro\triA{distancebetween("B","C")}
  \pgfmathsetmacro\triB{distancebetween("C","A")}
  \pgfmathsetmacro\triC{distancebetween("A","B")}
  \path (barycentric cs:A=\triA,B=\triB,C=\triC) coordinate (M)
  \draw (M.75-90) coordinate (@)
  \path (intersection of @--[shift=(vecB)]@ and B--C) coordinate (C) --
  \path (intersection of @--[shift=(vecB)]@ and B--A) coordinate (A);
}\end{tikzpicture}

anglebetween("p1", "p2")
\pgfmathanglebetween{"p1"}{"p2"}

Return the angle between the centers of the nodes $p1$ and $p2$.

qanglebetween("p")
\pgfmathqanglebetween{"p"}

Return the angle between the origin and the center of the node $p$.

distancebetween("p1", "p2")
\pgfmathdistancebetween{"p1"}{"p2"}

Return the distance (in pt) between the centers of the nodes $p1$ and $p2$.

qdistancebetween("p")
\pgfmathqdistancebetween{"p"}

Return the distance (in pt) between the origin and the center of the node $p$.

26.2 \texttt{pgffor}

This library loads also most of the functions of the \texttt{pgffor-ext} of section 25 on page 63.
26.3 **PGFkeys**

The `pgfkeys` library extends `pgf` keys and adds helpful `/utils` keys as well as handlers. This library gets loaded by the `ext.misc` library.

26.3.1 **Conditionals**

```
/utils/if={⟨cond⟩}{⟨true⟩}{⟨false⟩}  \ (no default)
```

This key checks the conditional `⟨cond⟩` and applies the styles `⟨true⟩` if `⟨cond⟩` is true, otherwise `⟨false⟩`. `⟨cond⟩` can be anything that `pgfmath` understands.

As a side effect on how `pgfkeys` parses argument, the `⟨false⟩` argument is actually optional.

The following keys use `\TeX`'s macros `\if`, `\ifx`, `\ifnum` and `\ifdim` for faster executions.

```
/utils/TeX/if=(token A)(token B){⟨true⟩}{⟨false⟩}  \ (no default)
```

This key checks via `\if` if `⟨token A⟩` matches `⟨token B⟩` and applies the styles `⟨true⟩` if it does, otherwise `⟨false⟩`. As a side effect on how `pgfkeys` parses argument, the `⟨false⟩` argument is actually optional.

```
/utils/TeX/ifx=(token A)(token B){⟨true⟩}{⟨false⟩}  \ (no default)
```

As above but via `\ifx`.

```
/utils/TeX/ifnum={⟨num cond⟩}{⟨true⟩}{⟨false⟩}  \ (no default)
```

This key checks `\ifnum⟨num cond⟩` and applies the styles `⟨true⟩` if true, otherwise `⟨false⟩`. A delimiting `\relax` will be inserted after `⟨num cond⟩`.

As a side effect on how `pgfkeys` parses arguments, the `⟨false⟩` argument is actually optional.

```
/utils/TeX/ifdim=(dim cond)(true)(false)  \ (no default)
```

As above but with `\ifdim`.

9`\usepgfkeyslibrary` is an upcoming feature of `pgf/TikZ`. For now, you need to load `ext.misc` or manually `\input` the file `pgfkeyslibraryext.pgfkeys-plus.code.tex` with `@` being a letter.

26.3.2 **Handlers**

While already a lot of values given to keys are evaluated by `pgfmath` at some point, not all of them are.

**Key handler** `⟨key⟩/.pgfmath=⟨eval⟩`

This handler evaluates `⟨eval⟩` before it is handed to the key.

This handler works almost the same as the `.evaluated` handler but it does its evaluation in a group so that the result will not overwrite any other results.

**Key handler** `⟨key⟩/.pgfmath int=⟨eval⟩`

As above but truncates the result.

**Key handler** `⟨key⟩/.pgfmath wrap=⟨wrapper⟩(⟨eval⟩)`

This feeds the result of `⟨eval⟩` as #1 to `⟨wrapper⟩`. In the example below, one could have used the `/pgf/foreach/evaluate` key from the `/foreach` loop.
Key handler (\texttt{key})/.pgfmath \texttt{if}=(\texttt{cond})\{\texttt{true}\}\{\texttt{false}\}

Evaluates \texttt{cond} with \texttt{pgfMath} and returns \texttt{true} or \texttt{false} to the used key respectively.

Key handler (\texttt{key})/.if=(\texttt{token A})(\texttt{token B})\{\texttt{true}\}\{\texttt{false}\}

Checks via \texttt{if} if \texttt{token A} matches \texttt{token B} and applies the value \texttt{true} if it does, otherwise \texttt{false}.

Key handler (\texttt{key})/.ifx=(\texttt{token A})(\texttt{token B})\{\texttt{true}\}\{\texttt{false}\}

As above but via \texttt{ifx}.

Key handler (\texttt{key})/.ifnum=(\texttt{ifnum cond})\{\texttt{true}\}\{\texttt{false}\}

Checks via \texttt{ifnum} if \texttt{ifnum cond} and applies the value \texttt{true} if it does, otherwise \texttt{false}.

Key handler (\texttt{key})/.ifdim=(\texttt{ifdim cond})\{\texttt{true}\}\{\texttt{false}\}

As above but via \texttt{ifdim}.

Key handler (\texttt{key})/.ifempty=(\texttt{Text})\{\texttt{true}\}\{\texttt{false}\}

Checks whether a fully expanded \texttt{Text} is empty and applies the value \texttt{true} if it does, otherwise \texttt{false}.

Key handler (\texttt{key})/.ifempty=(\texttt{Text})\{\texttt{true}\}\{\texttt{false}\}

Checks whether \texttt{Text} is empty and applies the value \texttt{true} if it does, otherwise \texttt{false}.

Key handler (\texttt{key})/.List=((\texttt{e1}), (\texttt{e2}), ..., (\texttt{en}))

This handler evaluates the given list with \texttt{foreach} and concatenates the element and the result is then given to the used key.
26.4 TikZ

/tikz/reverse clip=⟨direction⟩ (default counter clockwise)

This key installs a very big rectangle which is either constructed counter clockwise (like the circle path operation) or clockwise.

/tikz/clip rule=⟨direction⟩ (default even odd)

This key switches directly\(^{10}\) to the specified rule which is either even odd or nonzero. This corresponds to the \(\text{/tikz/even odd rule}\) and \(\text{/tikz/nonzero rule keys}\).

\(\text{\tiny \texttt{\textbackslashusetikzlibrary{ext.misc}}}
\text{\textbackslash newcommand*\{\myDiagram\[1\]}\{
\text{\textcolor{blue}{fill}}\left[\textcolor{green}{left color}, \textcolor{red}{right color}\right]\{0, 0\} \text{ rectangle } \{2, 1\};
\text{\textcolor{blue}{clip}} \{1, .5\} \#1 \text{ [reverse clip]};
\text{\textcolor{blue}{fill}}\left[\textcolor{green}{left color}, \textcolor{red}{right color}\right]\{0, 0\} \text{ rectangle } \{2, 1\};
\}
\text{\textbackslash begin\{tikzpicture\}[radius=.4, row sep=5\text{mm}, column sep=5\text{mm}]}
\text{\begin{array}{cc}
\text{\myDiagram\{circle\}} & \text{\myDiagram\{+(0:.4) arc[start angle=0, delta angle=-360] -- cycle\}} \\
\text{\myDiagram\{circle\}} & \text{\myDiagram\{+(0:.4) arc[start angle=0, delta angle=-360] -- cycle\}}
\end{array}}
\text{\end{tikzpicture}}\)

\(^{10}\)Meaning, it directly executes \texttt{\textbackslash pgfseteorule} /\texttt{\textbackslash pgfsetnonzerorule} and doesn’t accumulates where TikZ throws an error.
Part V
Changelog, Index & References

Changelog

Version 0.5.1
- Added PGF library ext.arrows.
- Bugfix to ext.pgfkeys-plus. [20]

Version 0.5
- Added package pgffor-ext.
- Added TikZ library ext.nodes.
- Added TikZ library ext.layers.
- Bugfixes to ext.calendar-plus.
- Allow the original rectangle timer with ext.paths.timer.

Version 0.4.2
- Added TikZ library ext.scalepicture.
- Bugfixes to shapes.uncenteredrectangle, paths.ortho, positioning-plus and pgfcalendar-ext.

Version 0.4.1
- Cleaned up directory structure of documentary.
- Added PGF keys library ext.pgfkeys-plus.
- Added shape uncetered rectangle (PGF library ext.shapes.uncenteredrectangle).
- Fixed ext.paths.arcto – again [19].

Version 0.4
- CTAN version of 0.3.1

Version 0.3.1
- Fixed ext.paths.ortho keys only vertical first and only horizontal first.
- Moved all (except the to paths) to namespace /tikz/ortho. /tikz/hvvh and /tikz/udlr are considered deprecated.
- Fixed \pgfcalendarjulianyartoweek.
- Added more calendar tests.
- Added directory structure.

Version 0.3
- Added shape circle arrow (PGF library ext.shapes.circlearrow).
- Added shape circle cross split (PGF library ext.shapes.circlecrosseplit).
- Added shape heatmark (PGF library ext.shapes.heatmark).
- Added shape rectangle with rounded corners (PGF library ext.shapes.rectangleroundedcorners).
- Added shape superellipse (PGF library ext.shapes.superellipse).
- Added TikZ library ext.node-families.shapes.geometric.
- Fixed ext.node-families`key size.
• Renamed internal macros to use custom namespace starting with `\tikzext@`.
• Added some references.

Version 0.2
• Added TiKZ library `ext.positioning-plus`.
• Added TiKZ library `ext.node-families`.

Version 0.1
• Added TiKZ library `ext.calendar-plus`.
• Added TiKZ library `ext.misc`.
• Added TiKZ library `ext.paths.arcto`.
• Added TiKZ library `ext.paths.ortho`.
• Added TiKZ library `ext.paths.timer`.
• Added TiKZ library `ext.patterns.images`.
• Added TiKZ library `ext.topaths.arcthrough`.
• Added TiKZ library `ext.transformations.mirror`.
• Added `pgf` library `ext.transformations.mirror`.
Index

This index contains automatically generated entries as well as references to original functionalities of PGF/Ti\LaTeX{}Z and references to functionalities outside of PGF/Ti\LaTeX{}Z.

- - path operation, 12
|-| path operation, 16
\-\- path operation, 16
\-\- path operation, 16
above key, 27
above left key, 25
above right key, 25
and date test, 61
angle between math function, 66
Apr date test, 61
arc path operation, 43
Arc Barb arrow tip, 38, 39
arc through key, 32
arc to path operation, 14
Arrow tips
  Arc Barb, 38, 39
  Bar, 38, 39
  Bracket, 38, 39
  Centered Arc Barb, 38, 39
  Centered Bar, 38
  Centered Bracket, 38
  Centered Circle, 38
  Centered Diamond, 38
  Centered Ellipse, 38
  Centered Hooks, 38
  Centered Kite, 38
  Centered Parenthesis, 38
  Centered Rays, 38
  Centered Rectangle, 38
  Centered Square, 38
  Centered Stealth, 38
  Centered Straight Barb, 38
  Centered Tee Barb, 38
  Centered Triangle, 38
  Centered Turned Square, 38
  Circle, 38, 39
  Diamond, 38
  Ellipse, 38, 39
  Hooks, 38
  Hug Cap, 39
  Kite, 38
  Loop, 39
  Parenthesis, 38, 39
  Rays, 38
  Rectangle, 38
  Square, 38
  Stealth, 38
  Straight Barb, 38
  Tee Barb, 38, 39
  Triangle, 38
  Turned Square, 38
  Untipped Bar, 39
  Untipped Bracket, 39
  Untipped Circle, 39
  Untipped Ellipse, 39
  Untipped Parenthesis, 39
  Untipped Tee Barb, 39
arrows.meta library, 37
atan2 math function, 41, 66
atanXY math function, 65
atanYX math function, 66
Aug date test, 61
background code key, 8
Bar arrow tip, 38, 39
below key, 27
below left key, 25
below right key, 25
Bracket arrow tip, 38, 39
calc library, 66
calendar library, 7
calendar week of month date test, 61
calendar week of month' date test, 61
calendar-ext package, 61
cd library, 57, 58
center suffix key, 32
Centered Arc Barb arrow tip, 38, 39
Centered Bar arrow tip, 38
Centered Bracket arrow tip, 38
Centered Circle arrow tip, 38
Centered Diamond arrow tip, 38
Centered Ellipse arrow tip, 38
Centered Hooks arrow tip, 38
Centered Kite arrow tip, 38
Centered Parenthesis arrow tip, 38
Centered Rays arrow tip, 38
Centered Rectangle arrow tip, 38
Centered Stealth arrow tip, 38
Centered Straight Barb arrow tip, 38
Centered Tee Barb arrow tip, 38
Centered Triangle arrow tip, 38
Centered Turned Square arrow tip, 38
Circle arrow tip, 38, 39
circle path operation, 69
circle shape, 9
circle arrow shape, 43
circle arrow key, 43
circle arrow delta angle key, 43
circle arrow end angle key, 43
circle arrow start angle key, 43
circle arrow turn left east key, 43
circle arrow turn left north key, 43
circle arrow turn left south key, 43
circle arrow turn left west key, 43
circle arrow turn right east key, 44
circle arrow turn right north key, 44
circle arrow turn right south key, 44
circle arrow turn right west key, 44
circle cross split shape, 46
circle cross split part fill key, 46
circle cross split uses custom fill key, 46
clip rule key, 69
clockwise key, 14, 32
code key, 8
corner above left key, 25
corner above right key, 25
corner below left key, 25
corner below right key, 25
corner east above key, 28
corner east below key, 28
corner north left key, 28
corner north right key, 28
corner south left key, 28
corner south right key, 28
corner west above key, 28
corner west below key, 28
cos math function, 41
cos path operation, 22
counter clockwise key, 15, 32

Date tests
and, 61
Apr, 61
Aug, 61
calendar week of month, 61
calendar week of month', 61
Dec, 61
Feb, 61
Jan, 61
Jul, 61
Jun, 61
leap year, 61
Mar, 61
May, 61
not, 61
Nov, 61
Oct, 61
Sep, 61
week, 61
week of month, 61
week of month', 61
yesterday, 61
day code key, 7
day text key, 7
day xshift key, 7
day yshift key, 7
Dec date test, 61
Diamond arrow tip, 38
distance key, 16
distancebetween math function, 66
du distance key, 18
east above key, 27
east below key, 27
dge path operation, 32
dge in box key, 8
dge on layer key, 8
Ellipse arrow tip, 38, 39
Environments pgfonlayer, 8
tikzcd, 58
evaluate key, 67
.evaluate handler, 67
even odd rule key, 69
every arc to key, 15
every day key, 7
every diagram key, 58
every month key, 7
every week key, 7
execute at end node key, 59
execute at end picture key, 9
ext.arrows library, 37
ext.calendar-plus library, 7
ext.layers library, 8
ext.misc library, 63–65
ext.node-families library, 9
ext.node-families.shapes.geometric library, 11
ext.nodes library, 12
ext.paths.arcto library, 14
ext.paths.ortho library, 16
ext.paths.timer library, 21
ext.patterns.images library, 24
ext.pgfkeys-plus pgfkeys library, 67
ext.positioning-plus library, 25
ext.scalepicture library, 30
ext.shapes.circlearrow library, 43
ext.shapes.circlecrosssplit library, 46
ext.shapes.heatmap library, 49
ext.shapes.rectangleangledcorners library, 52
ext.shapes.supereclipse library, 54
ext.shapes.uncenteredrectangle library, 57, 58
ext.topaths.arctrough library, 32
ext.transformations.mirror library, 33, 41
external library, 9
Feb date test, 61
fit library, 25
fit bounding box key, 29
foreground code key, 8
from center key, 17
full arc key, 65
heatmap shape, 49
heatmap arc (arc number) key, 50
heatmap arc rings key, 49
heatmap arc sep key, 49
heatmap arc sep angle key, 49
heatmap arc width key, 49
heatmap arcs key, 49
heatmap inner opacity key, 49
heatmap outer opacity key, 49
heatmap ring (ring number) key, 49
heatmap ring (ring number) arc (arc number) key, 50
height key, 10, 11
Hooks arrow tip, 38
horizontal vertical key, 19
horizontal vertical horizontal key, 19
Hug Cap arrow tip, 39
.if handler, 68
if key, 7, 67
\ifdate, 62
Key handlers

\item \List, 68
\item evaluated, 67
\item if, 68
\item ifdim, 68
\item ifempty, 68
\item ifnum, 68
\item ifx, 68
\item ifxempty, 68
\item .list xparse, 64
\item pgfmath, 67
\item pgfmath if, 68
\item pgfmath int, 67
\item pgfmath wrap, 67

Kite arrow tip, 38

large key, 15

lastdayofmonthofyear math function, 7

leap year date test, 61

Jan date test, 61

Jul date test, 61

Jun date test, 61

Libraries

\item arrows.meta, 37
\item calc, 66
\item calendar, 7
\item cd, 57, 58
\item ext.arrows, 37
\item ext.calendar-plus, 7
\item ext.layers, 8
\item ext.misc, 63–65
\item ext.node-families, 9
\item ext.node-families.shapes.geometric, 11
\item ext.nodes, 12
\item ext.paths.arcto, 14
\item ext.paths.ortho, 16
\item ext.paths.timer, 21
\item ext.patterns.images, 24
\item ext.positioning-plus, 25
\item ext.scalepicture, 30
\item ext.shapes.circlearrow, 43
\item ext.shapes.circlecrosssplit, 46
\item ext.shapes.heatmark, 49
\item ext.shapes.rectangleroundedcorners, 52
\item ext.shapes.superellipse, 54
\item ext.shapes.uncenteredrectangle, 57, 58
\item ext.topaths.arcthrough, 32
\item ext.transformations.mirror, 33, 41
\item external, 9
\item fit, 25
\item intersections, 13
\item markings, 12
\item positioning, 25
\item shapes.geometric, 11
\item spath3, 13
\item .List handler, 68
\item .list xparse handler, 64

Loop arrow tip, 39

lr distance key, 18

Mar date test, 61
mark connection node key, 12
markings library, 12
Math functions
anglebetween, 66
atan2, 41, 66
atanXY, 65
atanYX, 66
cos, 41
distancebetween, 66
isEmpty, 65
isInString, 65
lastdayinmonthofyear, 7
qanglebetween, 66
qdistancebetween, 66
sin, 41
strcat, 65
strrepeat, 65
superellipsex, 55
superellipsy, 55
weeksinmonthofyear, 7
math mode key, 58
Math operators
R, 65
matrix in box key, 8
matrix of math nodes key, 58
matrix on layer key, 8
maximum picture height key, 30
maximum picture size key, 30
maximum picture width key, 30
maximum picture width* key, 31
May date test, 61
middle 0 to 1 key, 18
minimum height key, 10, 29
minimum picture height key, 30
minimum picture size key, 30
minimum picture width key, 30
minimum picture width* key, 31
minimum width key, 10, 29
Mirror key, 35
mirror key, 34
Mirror x key, 35
mirror x key, 34
month code key, 7
month text key, 7
month xshift key, 7
month yshift key, 7
name key, 24
no separator key, 63
node in box key, 8
node on layer key, 8
node on line key, 12
nodes on curve key, 13
nodes on curve' key, 13
nodes on line key, 12, 13
nonzero rule key, 69
normal list key, 63
north left key, 26
north right key, 27
not date test, 61
Nov date test, 61
Oct date test, 61
on grid key, 29
on layer key, 8
only horizontal first key, 19
only horizontal second key, 19
only vertical first key, 19
only vertical second key, 19
option key, 24
options key, 24
outer sep key, 13

Packages and files
calendar-ext, 61
ggfor, 63
ggfor-ext, 63, 66
xparsp, 64
parabola path operation, 22
Parenthesis arrow tip, 38, 39
patch key, 8
Path operations
\hspace{.5em}--\hspace{.5em}, 12
\hspace{.5em}|-\hspace{.5em}, 16
\hspace{.5em}-|\hspace{.5em}, 16
\hspace{.5em}|-\hspace{.5em}, 16
\hspace{.5em}|\hspace{.5em}, 16
arc, 43
arc to, 14
circle, 69
cos, 22
dge, 32
parabola, 22
r-du, 18
r-lr, 18
r-rl, 18
r-ud, 18
rectangle, 21
sin, 22
to, 32

/pgf/
\hspace{.5em}arrow keys/
\hspace{.5em}length, 39
circle arrow arrows, 43
circle arrow delta angle, 43
circle arrow end angle, 43
circle arrow start angle, 43
circle arrow turn left east, 43
circle arrow turn left north, 43
circle arrow turn left south, 43
circle arrow turn left west, 43
circle arrow turn right east, 44
circle arrow turn right north, 44
circle arrow turn right south, 44
circle arrow turn right west, 44
circle cross split part fill, 46
circle cross split uses custom fill, 46
decoration/
\hspace{.5em}mark connection node, 12
foreach/
\hspace{.5em}evaluate, 67

no separator, 63
normal list, 63
use float, 63
use int, 63
var, 64
xparser, 64
xparser 0m, 64
full arc, 65
heatmark arc \{arc number\}, 50
heatmark arc rings, 49
heatmark arc sep, 49
heatmark arc sep angle, 49
heatmark arc width, 49
heatmark arcs, 49
heatmark inner opacity, 49
heatmark outer opacity, 49
heatmark ring \{ring number\}, 49
heatmark ring \{ring number\} arc \{arc number\}, 50
minimum height, 10, 29
minimum width, 10, 29
outer sep, 13
rectangle with rounded corners north east radius, 52
rectangle with rounded corners north west radius, 52
rectangle with rounded corners radius, 52
rectangle with rounded corners south east radius, 52
rectangle with rounded corners south west radius, 52
shape border rotate, 49
superellipse exponent, 54
superellipse step, 54
superellipse x exponent, 54
superellipse y exponent, 54
text, 24
uncentered rectangle center, 57, 59
uncentered rectangle center yshift, 57
uncentered rectangle use saved center, 57
\pgfcalendar, 62
\pgfcalendarcurrentweek, 62
\pgfcalendaridateweek, 62
\pgfcalendarjulianyearweek, 62
pgffor package, 63
pgffor-ext package, 63, 66

77
circle, 9
circle arrow, 43
circle cross split, 46
heatmark, 49
rectangle, 9
rectangle with rounded corners, 52
superellipse, 54
uncentered rectangle, 57
shapes.geometric library, 11
sin math function, 41
sin path operation, 22
size key, 10
small key, 15
south left key, 27
south right key, 27
spacing key, 17
span key, 29
span horizontal key, 29
span vertical key, 29
spath3 library, 13
Square arrow tip, 38
Stealth arrow tip, 38
Straight Barb arrow tip, 38
strcat math function, 65
strrepeat math function, 65
superellipse shape, 54
superellipse exponent key, 54
superellipse step key, 54
superellipse x exponent key, 54
superellipse y exponent key, 54
superellipses math function, 55
superellipses math function, 55

Tee Barb arrow tip, 38, 39
text key, 24
text key, 9
text depth key, 9
text height key, 9
text width key, 9
text width align key, 10
through key, 32

/tikz/
above, 27
above left, 25
above right, 25
arc through/
center suffix, 32
clockwise, 32
counter clockwise, 32
through, 32
arc through, 32
arc to/
clockwise, 14
counter clockwise, 15
large, 15
radius, 15
rotate, 15
small, 15
x radius, 15
y radius, 15
below, 27
below left, 25
below right, 25
clip rule, 69
commutative diagrams/
every diagram, 58
install uncentered rectangle in columns, 59
math mode, 58
corner above left, 25
corner above right, 25
corner below left, 25
corner below right, 25
corner east above, 28
corner east below, 28
corner north left, 28
corner north right, 28
corner south left, 28
corner south right, 28
corner west above, 28
corner west below, 28
day code, 7
day text, 7
day xshift, 7
day yshift, 7
east above, 27
east below, 27
edge in box, 8
every arc to, 15
every day, 7
every month, 7
every week, 7
execute at end node, 59
execute at end picture, 9
horizontal bounding box, 29
horizontal vertical, 19
horizontal vertical horizontal, 19
if, 7
image as pattern/
   name, 24
   option, 24
   options, 24
   image as pattern, 24
left, 26, 27
matrix in box, 8
matrix of math nodes, 58
matrix on layer, 8
maximum picture height, 30
maximum picture size, 30
maximum picture width, 30
maximum picture width*, 31
minimum picture height, 30
minimum picture size, 30
minimum picture width, 30
minimum picture width*, 31
Mirror, 35
mirror, 34
Mirror x, 35
mirror x, 34
Mirror y, 35
mirror y, 34
node family/
   height, 10, 11
   prefix, 9
   setup shape, 10
   size, 10
   text, 9
   text depth, 9
   text height, 9
   text width, 9
   text width align, 10
   width, 10, 11
node in box, 8
node on layer, 8
node on line, 12
nodes on curve, 13
nodes on curve', 13
nodes on line, 12, 13
node family/
   height, 10, 11
   prefix, 9
   setup shape, 10
   size, 10
   text, 9
   text depth, 9
   text height, 9
   text width, 9
   text width align, 10
   width, 10, 11
node in box, 8
node on layer, 8
node on line, 12
nodes on curve, 13
nodes on curve', 13
nodes on line, 12, 13
nonzero rule, 69
north left, 26
north right, 27
only horizontal first, 19
only horizontal second, 19
only vertical first, 19
only vertical second, 19
orthogonal
   distance, 16
   du distance, 18
   from center, 17
   install shortcuts, 20
   lr distance, 18
   middle 0 to 1, 18
   ratio, 16
   rl distance, 18
   spacing, 17
   ud distance, 18
   udlr distance, 18
pic, 12
pic in box, 8
pic on layer, 8
pics/
  background code, 8
code, 8
  foreground code, 8
picture height, 30
picture height*, 31
picture width, 30
picture width*, 30
pos, 17, 21
rectangle timer, 21
reverse clip, 69
right, 27
save picture size, 30
south left, 27
south right, 27
span, 29
span horizontal, 29
span vertical, 29
to path, 12, 19
uncrc, 59
uncrc math mode, 59
vertical horizontal, 19
vertical horizontal vertical, 19
week code, 7
week label left, 7
week text, 7
west above, 27
west below, 27
x radius, 15
xMirror, 35
xmirror, 33
y radius, 15
yMirror, 35
ymirror, 34
tikz-cd fix key, 58

\tikz\text{xext\text{}pictureheight}, 30
\tikz\text{xext\text{}picturewidth}, 30
to path operation, 32
to path key, 12, 19
Triangle arrow tip, 38
Turned Square arrow tip, 38
ud distance key, 18
udlr distance key, 18
uncentered rectangle shape, 57
uncentered rectangle center key, 57, 59
uncentered rectangle center yshift key, 57
uncentered rectangle use saved center key, 57
\uncrec, 59
uncrc key, 59
uncrc math mode key, 59
Untipped Bar arrow tip, 39
Untipped Bracket arrow tip, 39
Untipped Circle arrow tip, 39
Untipped Ellipse arrow tip, 39
Untipped Parenthesis arrow tip, 39
Untipped Tee Barb arrow tip, 39
use float key, 63
use int key, 63
\text{/utils/}
  if, 67
  TeX/
  \text{if}, 67
  ifdim, 67
  ifempty, 67
  ifnum, 67
  ifx, 67
  ifxempt, 67
var key, 64
vertical horizontal, 19
vertical horizontal vertical, 19
week date test, 61
week code key, 7
week label left key, 7
week of month date test, 61
week of month' date test, 61
week text key, 7
weeks in month of year math function, 7
west above key, 27
west below key, 27
width key, 10, 11
x radius key, 15
x radius key, 15
xMirror key, 35
xmirror key, 35
xparser package, 64
xparser key, 64
xparser 0m key, 64
y radius key, 15
y radius key, 15
yesterday date test, 61
yMirror key, 35
ymirror key, 34
References


[38] Qrrbrbirlbel. Answer to “Modifying * and o style tikz arrows so that they are centered at the end of line”. TeX - LaTeX Stack Exchange. Sept. 2022. URL: https://tex.stackexchange.com/a/656241 (visited on 04/02/2023) (cit. on p. 37).
