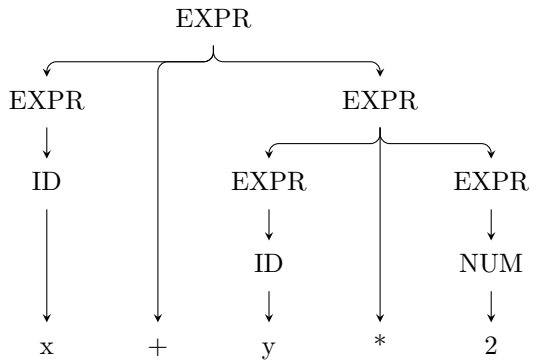
terminaltrees

Evenly spaced leaves, rounded branches

Adi Edelhaus

2026/10/28 v1.0.2 Child-centered terminal trees



`terminaltrees` adds a reusable layout to Forest. Terminal leaves sit at equal intervals on one baseline. Each parent is centered between its first and last immediate children. Side branches curve. Children directly below their parent connect straight down.

Label measurements determine spacing. An optional fitting environment shrinks the whole tree to the available page dimensions, subject to a minimum label size. Neither form chooses paper sizes or splits a tree across pages.

Contents

1 Install and draw a tree **2**

2 Keep the type size, or fit the page **2**

3 See what fitting changes **3**

4 Spacing, labels and curves **4**

5 How the layout is calculated **4**

6 Examples, checks and compatibility **5**

7 License, credits and support **5**

1 Install and draw a tree

With a current TeX Live installation, install the package using its package manager:

```
tlmgr install terminaltrees
```

If your operating system manages TeX Live, use its package manager instead.

With MiKTeX, refresh the package catalogue and install `terminaltrees`:

```
miktex packages update-package-database
miktex packages install terminaltrees
```

For manual installation, download the CTAN archive and place `terminaltrees.sty` beside your document, or upload it to your Overleaf project. For a personal TeX installation, place the file in `tex/latex/terminaltrees/` under your user TEXMF tree. Use your TeX distribution's instructions to refresh its filename database if required.

The package loads Forest's edges library and needspace. Forest loads PGF/TikZ. Do not copy these dependencies into your document folder.

Compile this complete example with pdfLaTeX:

```
\documentclass{article}
\usepackage{terminaltrees}
\begin{document}
\begin{forest}
  terminal tree
  [EXPR [EXPR [ID [x]]] [+]
    [EXPR [EXPR [ID [y]]] [*] [EXPR [NUM [2]]]]]
\end{forest}
\end{document}
```

It produces the tree on the cover. Labels use ordinary LaTeX and Forest bracket syntax.

2 Keep the type size, or fit the page

Both forms use Forest and the same layout:

`forest` with `terminal tree` Keeps the tree at its natural size. You manage placement and choose paper large enough for it. This form can be used inside figures and minipages.

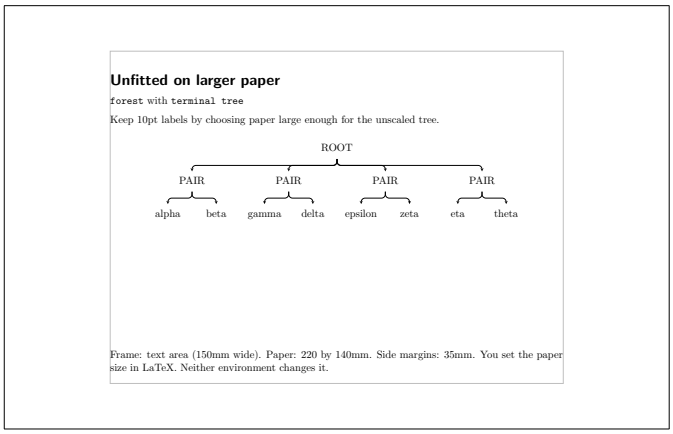
`fittedterminaltree` Shrinks a tree when necessary and reserves space in normal single-column document flow. If it will not fit in the remaining space, it moves to the next page. It does not enlarge small trees or change the paper or margins.

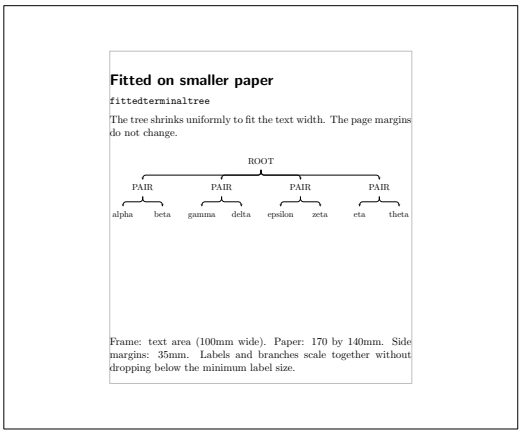
```
\begin{fittedterminaltree}
  [EXPR [EXPR [x]] [+] [EXPR [1]]]
\end{fittedterminaltree}
```

The fitting environment applies `terminal tree` automatically. Forest options go before the root bracket. Fitting options use *parentheses* after the environment name:

```
\begin{fittedterminaltree}(minimum label size=7pt)
  roundness=0.5,
  for tree={font=\small}
  [EXPR [EXPR [x]] [+] [EXPR [1]]]
\end{fittedterminaltree}
```

3 See what fitting changes





This comparison is generated from the included examples. Both paper previews share one reduction, so their paper and label sizes remain comparable. The example deliberately uses `leaf pitch=16mm`, not the package default.

The author chooses the larger or smaller paper in LaTeX. The unfitted tree keeps its original label size. The fitted tree scales labels and branches together within the smaller text area. The separate `fitting-comparison.pdf` also includes a third page showing deliberate unfitted overflow. Use the same zoom for its pages rather than fitting each page separately to the window.

3.1 Fitting and the font-size floor

The environment typesets the tree once, then chooses the largest uniform scale no greater than 1 that fits `\linewidth` and `\textheight - \baselineskip`. It rounds the scale down to TeX's supported precision and reserves space with `\Needspace*`. Moving to another page does not change the chosen scale.

`minimum label size` defaults to the document's current `\footnotesize` size. It must be positive. The test uses the smallest base font selected for a node, multiplied by the scale. Select node fonts through Forest's `font` option. Inline font changes, mathematical subscripts and externally scaled content are not measured. Do not replace the fitting environment's `execute at begin node` hook. The floor is an author-selected limit, not a guarantee of readability for every reader.

If the floor cannot be met, compilation stops. Wrap wide labels, increase a node font that is already below the floor, or split the tree manually. A PDF partially written by a failed compilation is not a valid result.

Fitting supports ordinary single-column flow, including lists. It rejects boxed contexts such as floats and minipages. Custom output routines, multicolumn layouts and automatic captions are outside the supported fitting interface. Use `forest` with `terminal tree` when managing placement yourself.

4 Spacing, labels and curves

4.1 Spacing

`leaf pitch` is the minimum center-to-center distance between consecutive leaves. Its default, `0pt`, imposes no additional minimum. Measurements of the labels and Forest's padding determine one uniform pitch for the tree. Wider labels or larger fonts therefore increase spacing automatically.

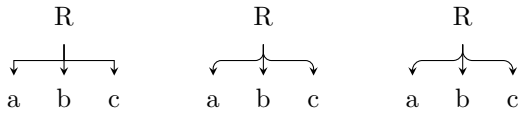
```
\begin{forest}
  terminal tree,
  leaf pitch=3em,
  for tree={font=\small}
  [ROOT [First\\label [a]] [Second [b]]]
\end{forest}
```

Use `\\` for deliberate line breaks. Forest controls vertical separation and multiline label height. Its label, font and arrow options remain available.

4.2 Roundness

`roundness` is a unitless number from 0 to 1. The default is 1, the maximum curvature permitted by each fork's geometry. Zero gives square corners. Intermediate values tighten the bends without moving nodes or changing fitting. Every tree starts at the default, independently of earlier trees.

`roundness=0` `roundness=0.5` `roundness=1`



The available space caps the radius. Sibling branches on the same side share a departure curve. Aligned branches stay straight. Values outside the range, physical units and non-finite values are fatal errors. PGF's math parser rejects malformed input.

5 How the layout is calculated

Leaves receive positions ip for indices $i = 0, \ldots, N-1$, where p is the effective pitch. Parent positions are computed from the leaves upward, each as the midpoint of its first and last immediate children. Equal leaf spacing does not imply equally spaced internal siblings. A parent need not be centered over all its descendant leaves. A middle branch is vertical only when its child and parent share the same horizontal position.

Each subtree reserves one column per descendant leaf, with boundaries half a leaf interval beyond its outer leaves. The pitch grows until each measured label box and half its Forest `s sep` on each side fit inside its subtree's boundaries. This conservative rule keeps unrelated labels and branches clear but does not minimize total width.

5.1 How branches are drawn

For each parent, the fork sits halfway through the clear space above its nearest child label. The shared bend radius is the smallest of:

- `roundness` times the fork distance
- half the shortest nonzero horizontal run to a child
- the fork distance minus the arrow clearance.

The result is bounded below by zero. Clearance uses PGF's shaft-shortening distance for the default thin stealth arrow, plus the outer padding at both anchors. This prevents the bend from reversing the arrow shaft. Forest's native forked-edge style draws the paths.

5.2 Custom styles and measurement limits

The style controls horizontal coordinates, terminal alignment, fork placement, corner radii and routing. Manual `fork sep` and `rounded corners` do not override the calculated geometry. Alternate growth directions, manual horizontal positioning and replacement paths are outside its intended use.

Automatic spacing assumes centered labels, nonnegative separation and positive vertical clearance. It does not measure shifted labels, custom anchors, overlays, edge labels, custom stroke extents or full custom arrowhead shapes. Inspect these additions visually. Shared path segments can be drawn more than once, so transparency and differing sibling styles also need inspection.

6 Examples, checks and compatibility

The archive includes example sources and the test suite. Run:

```
python3 terminaltrees-tests/check.py --preview
```

This writes renderings to a temporary directory and reports its location. It needs Python 3, pdfLaTeX, `pdfinfo`, `pdftotext`, `pdftocairo`, and the standalone class. The default checker also compares the README SVG byte-for-byte. Different TeX, fonts or Poppler versions can change that output without changing layout.

The package has been tested with pdfLaTeX on TeX Live 2023 (Ubuntu) and TeX Live 2026 (macOS). Other engines and Overleaf execution have not been verified. The package does not provide tagged-PDF tree semantics or screen-reader support.

7 License, credits and support

Copyright 2026 Adi Edelhaus. Licensed under the LaTeX Project Public License, version 1.3c or later. Maintenance status: maintained. Current maintainer: Adi Edelhaus. See `LICENSE` and `manifest.txt` in the archive.

Built on Forest by Sašo Živanović, PGF/TikZ, the LaTeX kernel and needspace. Dependencies retain their own licenses and are not bundled. Report issues at https://github.com/edel67/terminaltrees/issues.