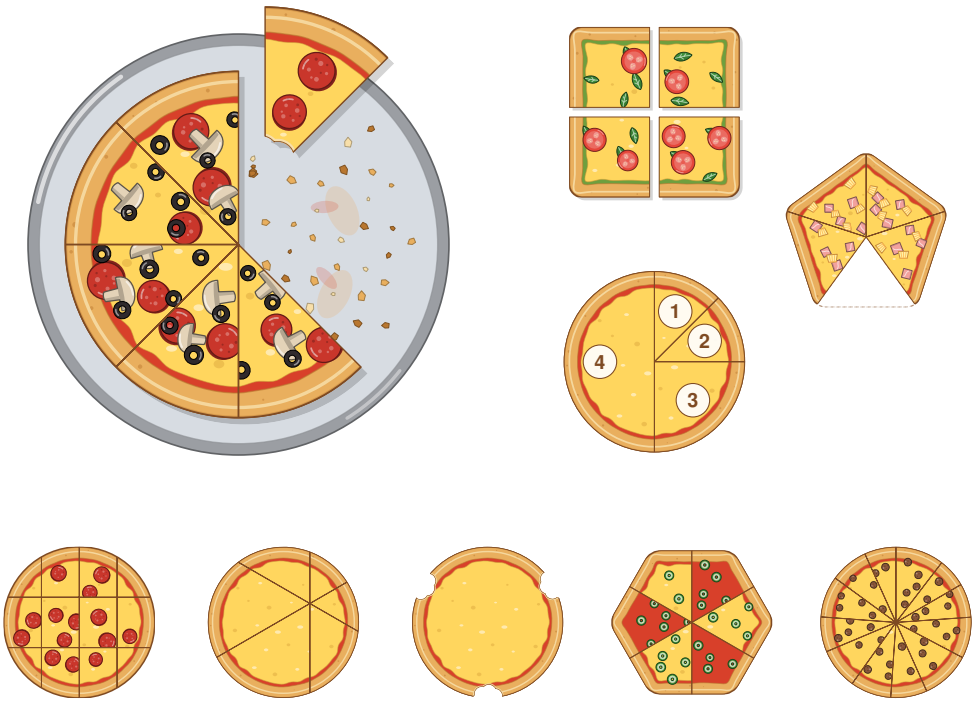
tikzpizza

Pizzas for fraction problems with Ti*k*Z



Vernon Merrick Fanning

Version 0.2 • 2026/11/04

Copyright © 2026 Vernon Merrick Fanning

Released under the LaTeX Project Public License 1.3c or later

Contents

1 Introduction **2**
- 1.1 Loading the package . . . 2

2 Basic use **2**

3 Shapes **3**

4 Cutting **4**
- 4.1 Equal slices . . . 4
- 4.2 Numbering and labels . . . 5
- 4.3 Slices of different sizes . . . 5
- 4.4 Grids and strips . . . 5
- 4.5 Off-centre cuts . . . 6
- 4.6 Sloppy cutting . . . 6
- 4.7 Hidden cuts . . . 7

5 Sauce, cheese and toppings **7**
- 5.1 Which slices . . . 7
- 5.2 Sauce and cheese . . . 7
- 5.3 The toppings . . . 8
- 5.4 Your own toppings . . . 9

6 Eating **10**
- 6.1 Slices that are gone . . . 10
- 6.2 Pulling slices out . . . 10
- 6.3 Bites . . . 10
- 6.4 The tray . . . 11

7 Fraction problems **12**

8 Colours **13**

9 Key reference **14**

10 Tips **15**

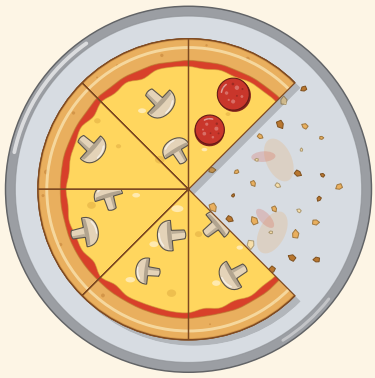
11 Change history **16**

12 Licence **16**

1 Introduction

tikzpizza draws pizzas for fraction problems. You choose the shape, how it is cut, what goes on each slice, and how much of it has been eaten; the package draws a cartoon pizza you can put on a worksheet, a test or a slide.

```
\begin{tikzpicture}
  \pizzatray[slices=8, eaten={2,3},
    pepperoni={on=1},
    mushroom={on=4-8}]
\end{tikzpicture}
```



Everything you would want for a fraction lesson can be set:

- the **shape**: round, square, rectangular or any regular polygon;
- the **cut**: into equal slices, into slices of any sizes you name, into a grid (the "tavern cut" on a round pizza, whose pieces are not all the same), or from an off-centre point, so that the fractions are no longer obvious;
- the **toppings**: sauce, cheese and fourteen toppings, each on exactly the slices you choose, as thick or as thin as you like, or an exact number per slice for counting;
- what is **gone**: slices eaten or taken away, slices pulled out of the pizza, bites out of the crust or the tip of a slice, and a tray underneath with the crumbs and smears that the missing slices left behind.

tikzpizza is a sibling of tikzeats, tikzfins and tikzbonds, and shares their conventions. All drawings are original, made from scratch with Ti*k*Z paths.

Until version 1.0, commands and keys may still change.

1.1 Loading the package

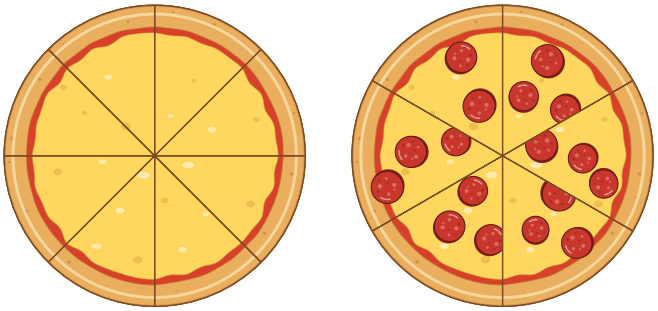
```
\usepackage{tikzpizza}
```

tikzpizza needs only Ti*k*Z, which it loads for you. It works with pdfLaTeX, XeLaTeX and LuaLaTeX, and it has no package options.

2 Basic use

`\pizza` draws a pizza inside a `tikzpicture`. Its optional argument takes ⟨*key*⟩=⟨*value*⟩ options:

```
\begin{tikzpicture}
  \pizza
  \pizza[at={(4.6,0)}, slices=6, pepperoni]
\end{tikzpicture}
```



By default a pizza is round, 2 cm in radius, centred on its `at` point, cut into eight equal slices, with tomato sauce and mozzarella and no toppings.

`at={(`⟨*x*⟩`,`⟨*y*⟩`)}`
: Moves the pizza. Put the coordinate in braces.

`scale=`⟨***factor***⟩
: Resizes it. Unlike Ti*k*Z's own `scale`, this key also scales the line widths.

`rotate=`⟨***angle***⟩
: Turns the whole pizza.

`line width=`⟨***factor***⟩
: Makes every line thicker or thinner (default 1).

`\pizzapic[`⟨*options*⟩`]` draws a pizza in the middle of a line of text, like this: . It is a `\tikz` picture around `\pizza`, aligned with the text.

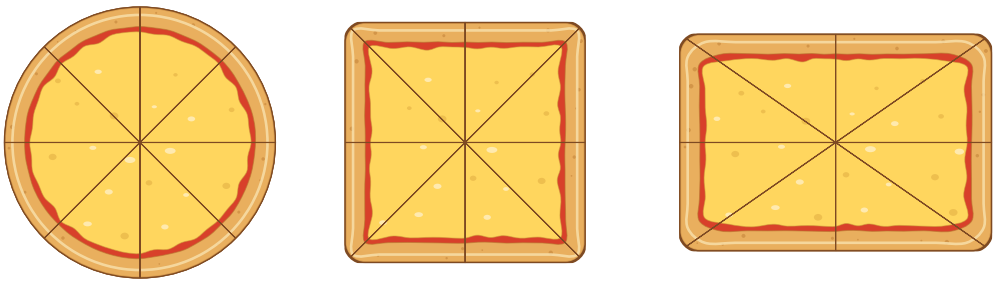
```
Ali ate \pizzapic[scale=0.15, slices=4, eaten=1] of a pizza.
```

`\pizzaset{`⟨*options*⟩`}` sets options for every later pizza in the current group, for instance a crust colour or a size for a whole worksheet:

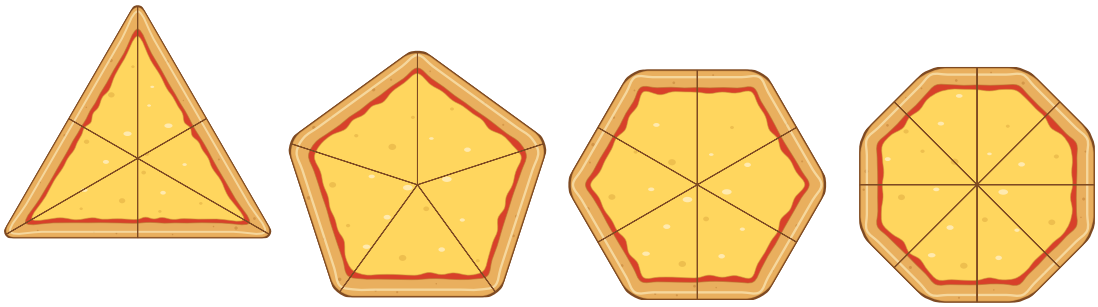
```
\pizzaset{scale=0.6, shape=square}
```

`\pizzatray` is `\pizza` with a tray underneath (section 6.4).

3 Shapes



`shape=circle` `shape=square` `shape=rectangle`



`shape=triangle` `shape=pentagon` `shape=hexagon` `shape=octagon`

`shape=`⟨*name*⟩
: `circle` (the default), `square`, `rectangle`, `triangle`, `pentagon`, `hexagon`, `octagon`, or `polygon` together with `sides`.

`sides=`⟨*n*⟩
: A regular polygon with ⟨*n*⟩ sides; it implies `shape=polygon`.

`radius=`⟨*length*⟩
: The size, default 2. A round pizza has this radius. A polygon is made with *as much pizza* as a round one of this radius: the same area, so pizzas of different shapes are fair to compare.

`side=`⟨*length*⟩
: Gives a polygon this side length instead.

`width`, `height`
: The size of a rectangle (default 4.6 by 3.2).

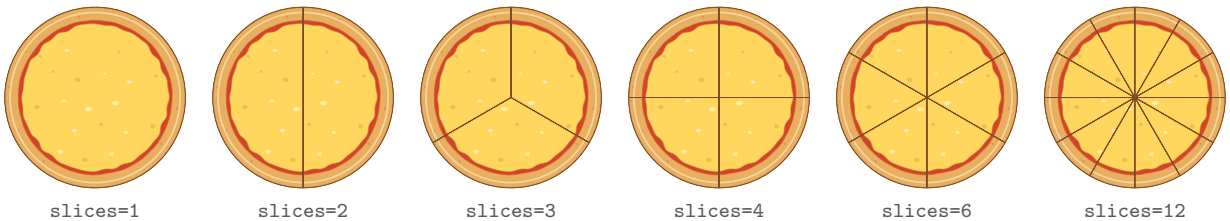
`corner radius=`⟨*length*⟩
: How round the corners of polygons and rectangles are (default 0.3; 0 gives sharp corners).

Polygons stand on a side, with a corner at the top when they have an odd number of sides. Use `rotate` to turn them.

4 Cutting

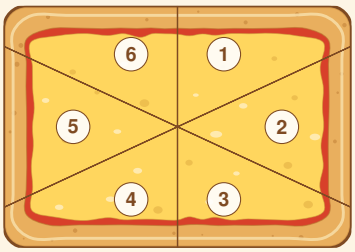
4.1 Equal slices

`slices=`⟨*n*⟩ cuts the pizza into ⟨*n*⟩ slices of the same size, from the centre. On a round pizza they have equal angles. On a polygon or a rectangle the cuts are placed so that every slice has the same *area*, even where that makes their angles different, so `slices=8` always means eighths. `slices=1` leaves the pizza whole.



`slices=1` `slices=2` `slices=3` `slices=4` `slices=6` `slices=12`

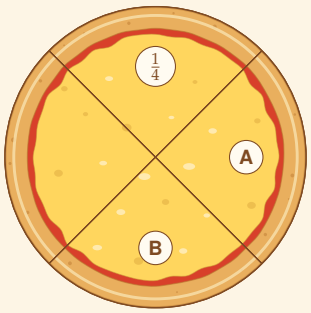
```
\begin{tikzpicture}
  \pizza[shape=rectangle, slices=6,
    numbers]
\end{tikzpicture}
```



4.2 Numbering and labels

Slices are numbered clockwise, starting with the slice just after the first cut. The first cut points straight up, so slice 1 starts at twelve o'clock. `start=`⟨*angle*⟩ moves the first cut (90 is up, 0 is to the right). `numbers` writes each slice's number on it, which is handy while you are deciding which slices get what, and for answer keys. `labels={`⟨*text*⟩`,`⟨*text*⟩`,...}` writes your own text on the slices instead, in order; leave an entry empty to skip a slice.

```
\begin{tikzpicture}
  \pizza[slices=4, start=45,
    labels={A,B,,$\frac14$}]
\end{tikzpicture}
```

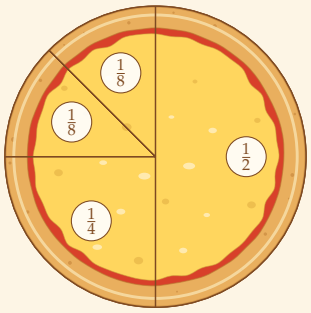


The labels use the Ti*k*Z style `pizza label`, which you can change with `\tikzset`.

4.3 Slices of different sizes

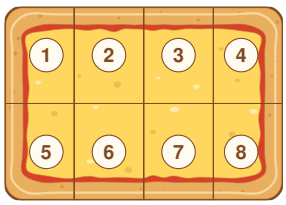
Give `slices` a list instead of a number, and the slices are cut in those proportions, clockwise from the first cut. The list can hold fractions or any numbers; only their ratios count, so `{1,1,2,4}` and `{1/8,1/8,1/4,1/2}` are the same pizza.

```
\begin{tikzpicture}
  \pizza[slices={1/2,1/4,1/8,1/8},
    labels={$\frac12$,$\frac14$,
            $\frac18$,$\frac18$}]
\end{tikzpicture}
```

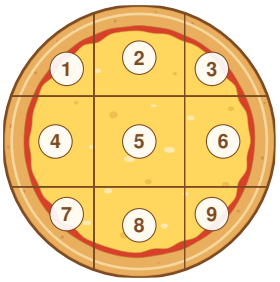


4.4 Grids and strips

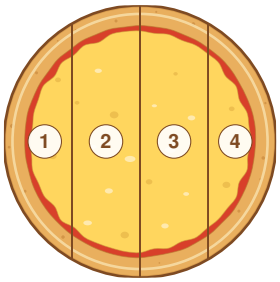
`grid=`⟨*columns*⟩`x`⟨*rows*⟩ cuts the pizza with straight lines into a grid of equal cells, like a sheet pizza; `grid=`⟨*n*⟩ is an ⟨*n*⟩ by ⟨*n*⟩ grid, and `strips=`⟨*n*⟩ cuts ⟨*n*⟩ vertical strips. On a square or rectangular pizza every piece is the same. On a round pizza this is the *tavern cut*: the middle pieces are whole squares, the edge pieces are not, and none of them is an obvious fraction.



shape=rectangle, grid=4x2



grid=3x3



strips=4

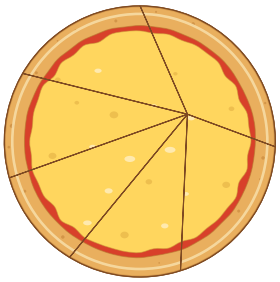
Grid pieces are numbered row by row, from the top left. Where a very fine grid leaves a corner cell that misses the pizza, the cell keeps its number but has nothing in it; slivers too thin to hold a label are not labelled.

4.5 Off-centre cuts

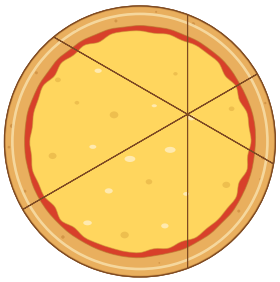
`hub={(`⟨*x*⟩`,`⟨*y*⟩`)}` makes the cuts meet somewhere other than the centre. What happens next depends on `cut by`:

`cut by=area` **(the default)**
: The slices still have the sizes you asked for. They *look* very different, but they are fair: a good puzzle.

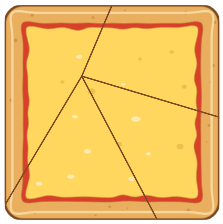
`cut by=angle`
: The cuts leave the hub at equal angles, as if someone had cut from the wrong place. The slices are *not* equal, and their sizes are not simple fractions.



hub={(0.7,0.4)}



hub={(0.7,0.4)}, cut by=angle



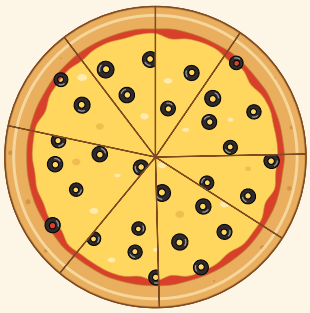
shape=square, hub={(-0.5,0.6)}

`cut by=angle` also works without a hub: a square cut into six by equal angles has slices of two different sizes.

4.6 Sloppy cutting

`sloppy=`⟨*amount*⟩ nudges every cut but the first a little, as a real pizza cutter would, so slices that ought to be equal are not quite. The amount runs from 0 (the default) to 1; a plain `sloppy` is 0.6. `seed` gives a different sloppy pizza.

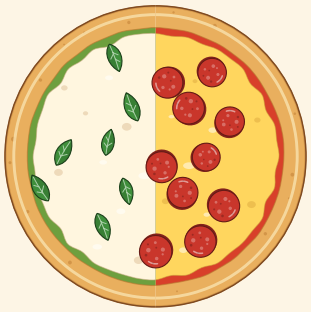
```
\begin{tikzpicture}
  \pizza[slices=8, sloppy=1, seed=3,
    olive]
\end{tikzpicture}
```



4.7 Hidden cuts

`cut lines=false` draws no cuts. The pizza still has its slices: they decide where the toppings go and where bites can be taken. This gives, for instance, an uncut half-and-half pizza:

```
\begin{tikzpicture}
  \pizza[slices=2, cut lines=false,
    sauce={pesto, on=2},
    cheese={white, on=2},
    basil={on=2}, pepperoni={on=1}]
\end{tikzpicture}
```

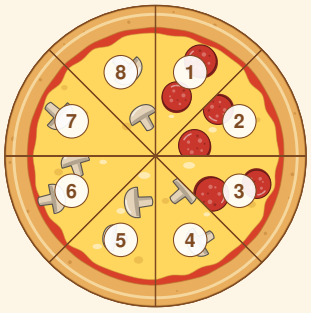


5 Sauce, cheese and toppings

5.1 Which slices

Every sauce, cheese and topping takes `on=`⟨*slices*⟩, a list of the slices it covers. Lists can contain single numbers, ranges like `2-6` or `3-last`, and the words `all` (the default), `none`, `odd`, `even`, `first` and `last`. Put a list with commas in braces: `on={1,3,5}`.

```
\begin{tikzpicture}
  \pizza[slices=8, numbers,
    pepperoni={on={1,2,3}},
    mushroom={on=4-last}]
\end{tikzpicture}
```

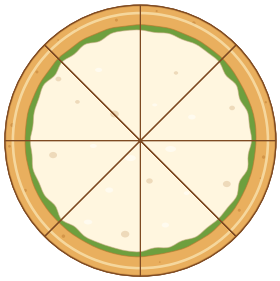


5.2 Sauce and cheese

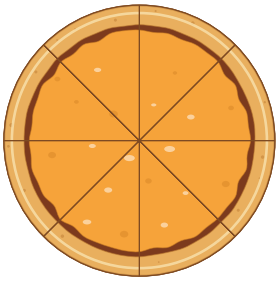
`sauce=`⟨*kind*⟩ is `tomato` (the default), `pesto`, `white`, `bbq` or `none` (bare dough). `cheese=`⟨*kind*⟩ is `mozzarella` (the default), `white` (fresh mozzarella), `cheddar` or `none`. Either can also take `color=`⟨*colour*⟩ and `on`:

```
\pizza[sauce=bbq, cheese=cheddar]
\pizza[sauce={pesto, on=1-4}, cheese={none, on=odd}]
```

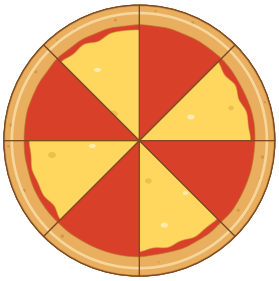
A sauce or cheese given later wins on the slices it covers, so you start from tomato and mozzarella everywhere and change only what is different. To put sauce on only some slices, take it away first: `sauce=none, sauce={tomato, on=1-4}`.



`sauce=pesto, cheese=white`



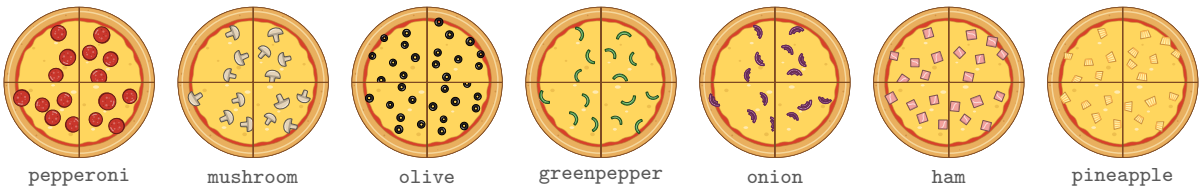
`sauce=bbq, cheese=cheddar`

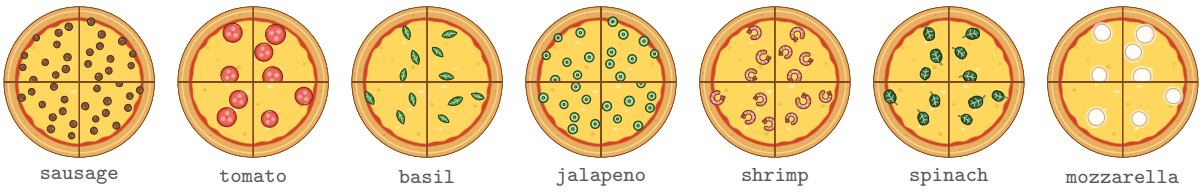


`cheese={none, on=odd}`

`crust=`⟨*colour*⟩, `crust width=`⟨*length*⟩ (default 0.3), `dough=`⟨*colour*⟩ and `outline=`⟨*colour*⟩ change the rest.

5.3 The toppings





Each topping is a key (`pepper` is another name for `greenpepper`). Used plainly it covers the whole pizza; with options in braces it takes:

`on=`⟨***slices***⟩
: Where it goes.

`density=`⟨***factor***⟩
: How thickly it is spread: 1 is an ordinary pizza, 2 twice as many, 0.5 half as many.

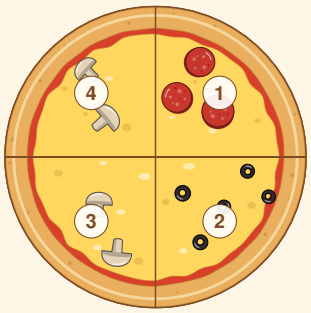
`count=`⟨***n***⟩
: Exactly ⟨*n*⟩ on *each* chosen slice, whole and not touching, so that they can be counted. If they do not fit, you get as many as fit and a warning.

`size=`⟨***factor***⟩
: Makes each piece bigger or smaller.

`color=`⟨***colour***⟩
: Recolours it.

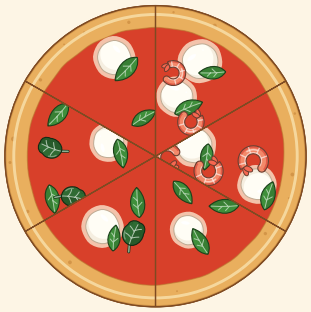
`seed=`⟨***n***⟩
: Scatters it differently.

```
\begin{tikzpicture}
  \pizza[slices=4, numbers,
    pepperoni={count=3, on=1},
    olive={count=5, on=2},
    mushroom={count=2, on=3-4}]
\end{tikzpicture}
```



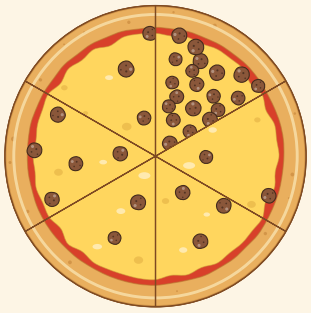
The `mozzarella` topping (also called `fancy mozzarella` or `fresh mozzarella`) is rounds of fresh mozzarella laid on top. It is not the same as `cheese=mozzarella`, the melted layer underneath: for a Margherita, take the melted cheese away and use the rounds with basil (here with shrimp and spinach on a few slices as well).

```
\begin{tikzpicture}
  \pizza[slices=6, cheese=none,
    fancy mozzarella, basil,
    shrimp={on=1-2},
    spinach={on=4-5}]
\end{tikzpicture}
```



Spread toppings (without `count`) are scattered evenly but not on a grid. They may run over a cut, where the cut slices through them, as on a real pizza. The same key can be used more than once, for instance thick on one slice and thin on the rest:

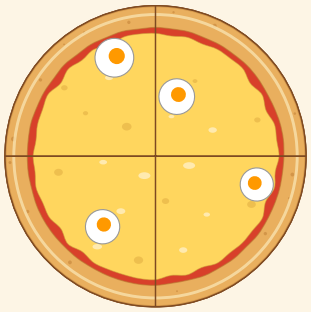
```
\begin{tikzpicture}
  \pizza[slices=6,
    sausage={on=1, density=3},
    sausage={on=2-6, density=0.4}]
\end{tikzpicture}
```



5.4 Your own toppings

`\pizzanewtopping{`⟨*name*⟩`}{`⟨*radius*⟩`}{`⟨*density*⟩`}{`⟨*colour*⟩`}{`⟨*drawing*⟩`}` adds a topping. Draw one piece around `(0,0)`, at its real size in centimetres, in the colour `pizzatopping`; use `\pizzalw` for thin lines. The ⟨*radius*⟩ is about how far the piece reaches from its centre (it keeps toppings off the crust and, in `count` mode, apart), and the ⟨*density*⟩ is how many pieces go on each square centimetre at `density=1`. The new topping takes all the options above.

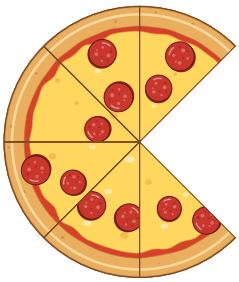
```
\pizzanewtopping{egg}{0.24}{0.8}{white}{%
  \fill[pizzatopping] (0,0)
    circle [radius=0.24];
  \fill[orange!80!yellow] (0.03,0.02)
    circle [radius=0.1];
  \draw[black!40, line width=\pizzalw]
    (0,0) circle [radius=0.24];}
\begin{tikzpicture}
  \pizza[slices=4, egg={count=1}]
\end{tikzpicture}
```



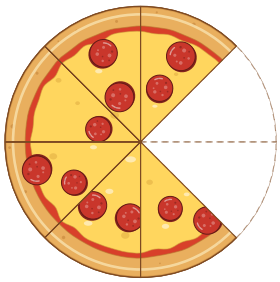
6 Eating

6.1 Slices that are gone

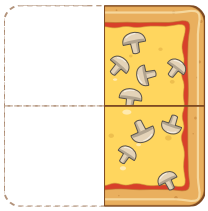
`eaten=`⟨*slices*⟩ leaves those slices out; `removed` and `missing` are other names for the same key, whichever reads better in your question. `ghost` draws a dashed outline where they were, to show what the whole was.



`eaten={2,3}`



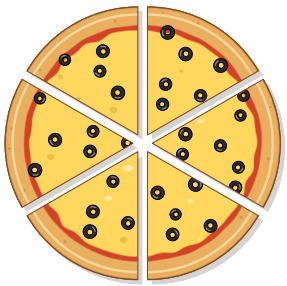
`eaten={2,3}, ghost`



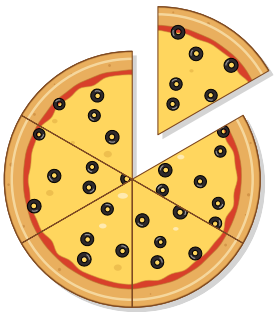
`eaten=3-last, ghost`

6.2 Pulling slices out

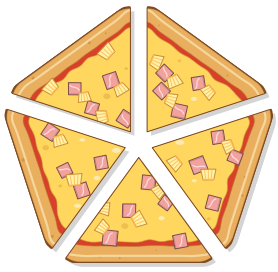
`separate` moves every slice a little away from the others (a plain `separate` is 0.15; give a distance for more), and `pull=`⟨*slices*⟩ slides those slices right out of the pizza, by the `pull distance` (default 0.8). Separated and pulled slices cast a small shadow.



`separate`



`pull=1`



`shape=pentagon, separate=0.25`

6.3 Bites

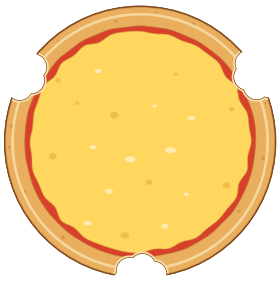
`bites=`⟨*n*⟩ takes ⟨*n*⟩ bites out of the pizza (a plain `bites` is one bite). A bite leaves a scalloped edge with the bread showing.

`bite on=`⟨***slices***⟩
: Which slices may be bitten (default `all`). The bites are shared out evenly among them, going round the pizza; when there are more bites than slices, some slices get several.

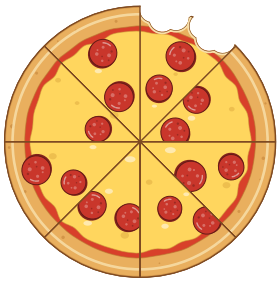
`bite from=edge`
: (the default) bites the crust.

`bite from=tip`
: bites a slice from its point, the way slices are eaten; more bites work their way towards the crust.

`bite size=`⟨***radius***⟩
: How big a mouthful is (default 0.42).



`slices=1, bites=3`



`bites=2, bite on=1`



`bites=2, bite from=tip`

6.4 The tray

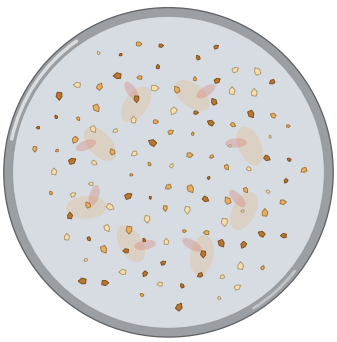
`tray` puts a tray under the pizza, and `\pizzatray` is `\pizza` with `tray` switched on. Where slices have been eaten or taken, the tray keeps what they left behind: crumbs, a smear of sauce and a little grease. Bites leave crumbs too.

`tray=`⟨***kind***⟩
: `round` (a pizza pan), `sheet` (a baking sheet) or `board` (a wooden board with a handle). A plain `tray` picks a sheet for square and rectangular pizzas and a pan for the rest.

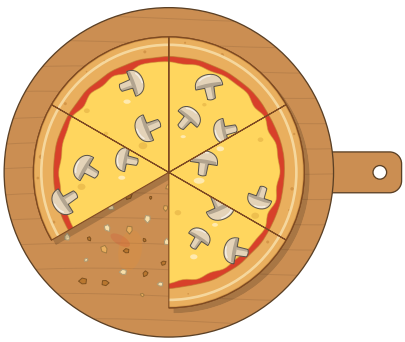
`crumbs=`⟨***factor***⟩
: How messy it is: 0 for no crumbs, 2 for twice as many (default 1).

`tray color=`⟨***colour***⟩
: The colour of a pan or sheet.

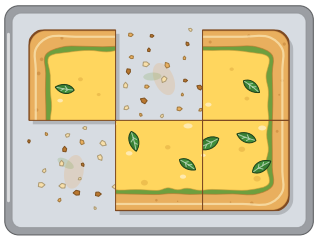
`shadow=`⟨***choice***⟩
: `auto` (the default) gives slices a shadow on a tray and when they are separated or pulled out; `true` and `false` switch it on or off everywhere.



`\pizzatray[eaten=all]`



`tray=board, eaten=4`



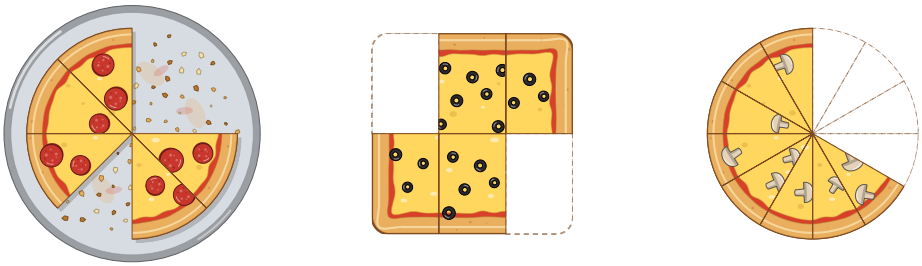
`shape=rectangle, tray`

7 Fraction problems

A few questions the package was made for. Draw the pizzas, then ask away.

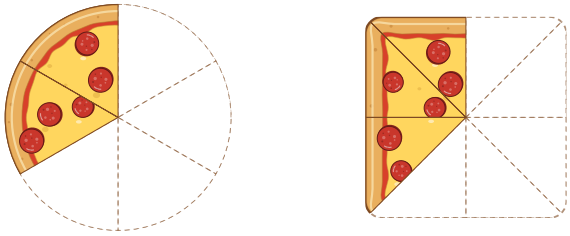
What fraction is left? Eaten slices, with or without the ghost of the whole:

```
\begin{tikzpicture}
  \pizzatray[slices=8, eaten={1,2,5}, pepperoni,
    scale=0.7]
  \pizza[at={(4.5,0)}, shape=square, grid=3x2,
    eaten={1,6}, ghost, scale=0.75, olive]
  \pizza[at={(9,0)}, slices=12, eaten=1-4, ghost,
    scale=0.7, mushroom]
\end{tikzpicture}
```



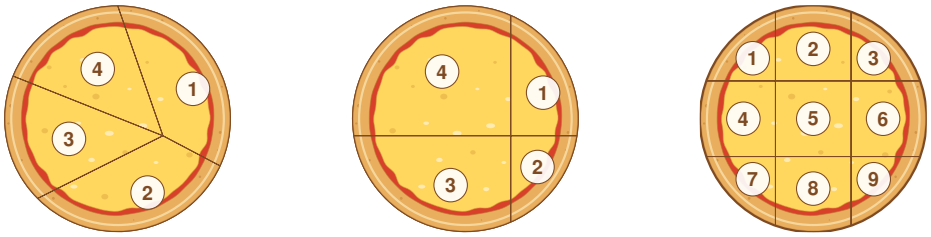
Who has more? Two friends order the same amount of pizza, cut differently. Polygons have the same area as a round pizza of the same `radius`, so the comparison is fair.

```
\begin{tikzpicture}
  \pizza[slices=6, eaten=1-4, ghost, scale=0.75,
    pepperoni]
  \pizza[at={(4.6,0)}, shape=square, slices=8, eaten=1-5,
    ghost, scale=0.75, pepperoni]
\end{tikzpicture}
```



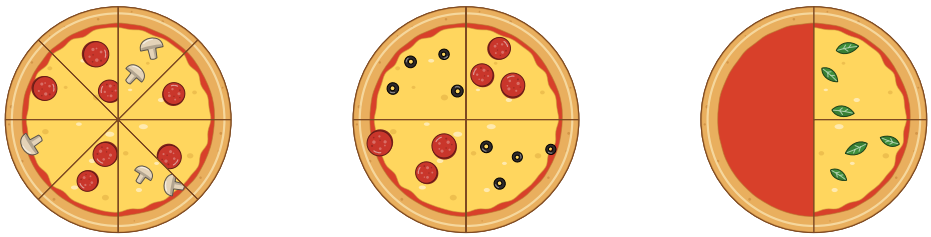
Is it fair? Cut from the wrong place, by area (fair) or by angle (not):

```
\begin{tikzpicture}
  \pizza[hub={(0.8,-0.3)}, slices=4, numbers, scale=0.75]
  \pizza[at={(4.6,0)}, hub={(0.8,-0.3)}, cut by=angle,
    slices=4, numbers, scale=0.75]
  \pizza[at={(9.2,0)}, grid=3x3, numbers, scale=0.75]
\end{tikzpicture}
```



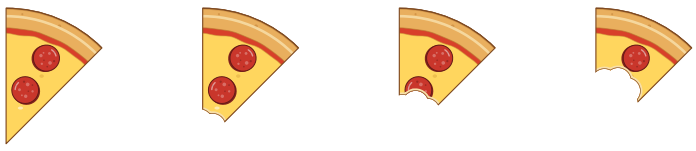
What fraction has mushrooms? Toppings on chosen slices, counted toppings for the youngest:

```
\begin{tikzpicture}
  \pizza[slices=8, mushroom={on={1,4,6}},
    pepperoni={on={2,3,5,7,8}}, scale=0.75]
  \pizza[at={(4.6,0)}, slices=4, pepperoni={count=3, on={1,3}},
    olive={count=4, on={2,4}}, scale=0.75]
  \pizza[at={(9.2,0)}, slices={1,1,2}, cheese={none, on=3},
    basil={on=1-2}, scale=0.75]
\end{tikzpicture}
```



How many bites? A slice eaten from the tip, one bite at a time:

```
\begin{tikzpicture}
  \foreach \b [count=\i from 0] in {0,1,2,3}{
    \pizza[at={(\i*2.6,0)}, slices=8, eaten=2-last, bites=\b,
      bite from=tip, pepperoni={count=2, on=1}, scale=0.9]
  }
\end{tikzpicture}
```



8 Colours

The package defines these named colours, which you can use anywhere:

pizzacrust
pizzadough
pizzabbq
pizzapepperoni
pizzaonion
pizzatomatoslice
pizzaspinach
pizzaboard
pizzacrustdark
pizzatomato
pizzamozzarella
pizzamushroom
pizzaham
pizzabasil
pizzamozzslice
pizzacrustlight
pizzapesto
pizzafreshmozz
pizzaolive
pizzapineapple
pizzajalapeno
pizzaseed
pizzaline
pizzawhite
pizzacheddar
pizzagreenpepper
pizzasausage
pizzashrimp
pizzatray

9 Key reference

Key	Values (default)	Notes
Placement		
`at`	coordinate (`(0,0)`)	The centre of the pizza.
`scale`	number (1)	Also scales line widths.
`rotate`	angle (0)	
`line width`	factor (1)	All lines.
Shape		
`shape`	`circle`, `square`, `rectangle`, `triangle`, `pentagon`, `hexagon`, `octagon`, `polygon` (`circle`)	
`sides`	number (5)	Implies `shape=polygon`.
`radius`	length (2)	Polygons get the same area as a circle of this radius.
`side`	length	A polygon's side, instead of `radius`.
`width, height`	lengths (4.6, 3.2)	Rectangles.
`corner radius`	length (0.3)	
Cutting		
`slices`	number or list (8)	A list gives slices of those relative sizes.
`grid`	⟨*c*⟩`x`⟨*r*⟩ or ⟨*n*⟩	Straight cuts into a grid.
`strips`	number	Vertical strips.
`start`	angle (90)	Where the first cut points.
`hub`	coordinate (`(0,0)`)	Where the cuts meet.
`cut by`	`area`, `angle` (`area`)	
`sloppy`	0–1 (0)	Plain `sloppy` is 0.6.
`seed`	integer (1)	A different scattering and sloppiness.
`cut lines`	`true`, `false` (`true`)	
`numbers`	`true`, `false` (`false`)	Number every slice.
`labels`	list of text	Text on each slice, in order.
Sauce, cheese and toppings		
`sauce`	`tomato`, `pesto`, `white`, `bbq`, `none`	Also `on`, `color`.
`cheese`	`mozzarella`, `white`, `cheddar`, `none`	Also `on`, `color`.
⟨*topping*⟩	options	`pepperoni`, `mushroom`, `olive`, `greenpepper` (`pepper`), `onion`, `ham`, `pineapple`, `sausage`, `tomato`, `basil`, `jalapeno`, `shrimp`, `spinach`, `mozzarella` (`fancy mozzarella`, `fresh mozzarella`), and your own.

Key	Values (default)	Notes
`on`	slice list (`all`)	Numbers, ranges, `all`, `none`, `odd`, `even`, `first`, `last`.
`density`	factor (1)	
`count`	number	Exactly this many on each slice.
`size`	factor (1)	
`color`	colour	
`seed`	integer (0)	
`crust, dough, outline`	colour	
`crust width`	length (0.3)	
Eating		
`eaten, removed, missing`	slice list (`none`)	
`ghost`	`true`, `false` (`false`)	Outline the missing slices.
`separate`	distance (0)	Plain `separate` is 0.15.
`pull`	slice list (`none`)	
`pull distance`	distance (0.8)	
`bites`	number (0)	Plain `bites` is 1.
`bite on`	slice list (`all`)	
`bite from`	`edge`, `tip` (`edge`)	
`bite size`	radius (0.42)	
Tray		
`tray`	`round`, `sheet`, `board`	Plain `tray` picks one for the shape.
`crumbs`	factor (1)	
`tray color`	colour	
`shadow`	`auto`, `true`, `false` (`auto`)	

10 Tips

Braces
Coordinates, and lists with commas, must be in braces:

```
\pizza[at={(3,0)}, hub={(0.5,0.2)}, eaten={2,5},
  pepperoni={on={1,3}, count=2}]
```

Finding a slice
Turn on `numbers` while you build a question, then take it off.

Same scattering
Toppings and crumbs are placed by a fixed rule, not at random, so a pizza looks the same every time you compile; change `seed` for a different arrangement.

Scaling the whole picture
The package's `scale` key scales line widths with the drawing. If you scale a whole picture with Ti*k*Z's own `scale`, the lines keep their width.

Compile time
A pizza takes about a third of a second, a little more with a tray full of crumbs or many toppings. A worksheet full of pizzas compiles in a few seconds; externalise finished pictures if you need it faster.

11 Change history

v0.2 (2026/11/04)

Three new toppings: `shrimp`, `spinach` and rounds of fresh `mozzarella` (also called `fancy mozzarella` and `fresh mozzarella`), with their colours `pizzashrimp`, `pizzaspinach` and `pizzamozzslice`.

v0.1 (2026/11/01)

First draft: round, rectangular and polygonal pizzas; equal, uneven, grid, off-centre and sloppy cuts; sauce, cheese and eleven toppings on chosen slices, by density or by count; eaten, ghost, separated and pulled slices; bites from the edge or the tip; trays with crumbs.

12 Licence

Copyright © 2026 Vernon Merrick Fanning.

This work may be distributed and/or modified under the conditions of the LaTeX Project Public License, either version 1.3c of this license or (at your option) any later version. The latest version of this license is in https://www.latex-project.org/lppl.txt and version 1.3c or later is part of all distributions of LaTeX version 2008 or later.

This work has the LPPL maintenance status "maintained". The Current Maintainer of this work is Vernon Merrick Fanning.

This work consists of the files `tikzpizza.sty`, `tikzpizza-doc.tex` and `README.md`, and the derived file `tikzpizza-doc.pdf`.

All drawings in this work are original. They were made from scratch with Ti*k*Z paths for this package; none is traced from or based on a photograph or on anyone else's artwork.