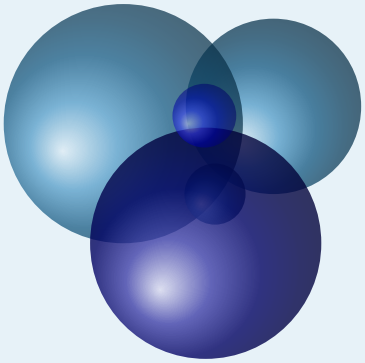
tkz-euclide 5.15c

AlterMundus

tkz-euclide v5

Euclidean Geometry



Alain Matthes

November 26, 2026 Documentation V.5.15c

http://altermundus.fr

tkz-euclide

AlterMundus

Alain Matthes

☞ **tkz-euclide** is a set of convenient macros for drawing figures in the plane with a Cartesian coordinate system. It handles the most classic situations of Euclidean geometry. **tkz-euclide** is built on top of PGF and its associated front-end Ti*k*Z and is a (La)TeX-friendly drawing package. It provides a high-level interface to build figures with little effort: you construct them step by step, as closely as possible to a construction made by hand.
Version 5 of **tkz-euclide** can use Lua for some calculations; see the sections `News` and `Lua` at the end of this documentation.
Please note: English is not my native language, so there may be some errors.

☞ First of all, I would like to thank **Till Tantau** for the beautiful package Ti*k*Z.

☞ Acknowledgements: I received much valuable advice, remarks, corrections and examples from `Jean-Côme Charpentier`, `Josselin Noirel`, `Manuel Pégourié-Gonnard`, `Franck Pastor`, `David Arnold`, `Ulrike Fischer`, `Stefan Kottwitz`, `Christian Tellechea`, `Nicolas Kisselhoff`, `Wolfgang Büchel`, `John Kitzmiller`, `Dimitri Kapetas`, `Gaétan Marris`, `Mark Wibrow`, `Yves Combe`, `Paul Gaborit`, `Laurent Van Deik` and `Muzimuzhi Z`.

☞ I would also like to thank Eric Weisstein, creator of MathWorld: MathWorld.

☞ You can find some examples on my site: altermundus.fr. under construction!

Please report typos or any other comments about this documentation to Alain Matthes.
This file can be redistributed and/or modified under the terms of the LaTeX Project Public License, distributed from CTAN archives.

Contents

I. Getting started 9

1. Presentation and overview 10
1.1. Why tkz-euclide? . . . 10
1.2. TikZ vs tkz-euclide . . . 10
1.3. How to use the tkz-euclide package? . . . 14

2. Installation 20

3. The Elements of tkz code 21
3.1. Objects and language . . . 21
3.2. Notations and conventions . . . 21
3.3. Set, Calculate, Draw, Mark, Label . . . 23

4. About this documentation and the examples 23

II. Points 24

5. First step: fixed points 25

6. Definition of a point: \tkzDefPoint or \tkzDefPoints 25
6.1. Defining a named point: \tkzDefPoint . . . 26
6.2. Point relative to another: \tkzDefShiftPoint . . . 27
6.3. Definition of multiple points: \tkzDefPoints . . . 29
6.4. Create a triangle . . . 29
6.5. Create a square . . . 29

7. Auxiliary tools 29
7.1. Constants . . . 29
7.2. New point by calculation . . . 30

8. Special points 30
8.1. Midpoint of a segment \tkzDefMidPoint . . . 31
8.2. Golden ratio \tkzDefGoldenRatio . . . 31
8.3. Barycentric coordinates with \tkzDefBarycentricPoint . . . 32
8.4. Internal and external similitude centers: \tkzDefSimilitudeCenter . . . 33
8.5. Harmonic division with \tkzDefHarmonic . . . 35
8.6. Equidistant points with \tkzDefEquiPoints . . . 36
8.7. Midpoint of an arc . . . 36

9. Point on line or circle 38
9.1. Point on a line with \tkzDefPointOnLine . . . 38
9.2. Point on a circle with \tkzDefPointOnCircle . . . 39

10. Special points relating to a triangle 41
10.1. Triangle center: \tkzDefTriangleCenter . . . 41

11. Definition of points by transformation 46
11.1. \tkzDefPointBy . . . 47
11.2. Transformation of multiple points: \tkzDefPointsBy . . . 55

12. Defining points using a vector 56
12.1. \tkzDefPointWith . . . 56
12.2. \tkzGetVectxy . . . 61

13. Random points 61
13.1. Obtaining random points . . . 61

III. Lines, triangles and polygons 63

14. Straight lines 64
14.1. Definition of straight lines . . . 64

15. Triangles 69
15.1. Definition of triangles \tkzDefTriangle . . . 69
15.2. Specific triangles with \tkzDefSpcTriangle . . . 75
15.3. Permutation of two points of a triangle . . . 82

16. Definition of polygons 83
16.1. Defining the points of a square . . . 83
16.2. Defining the points of a rectangle . . . 84
16.3. Definition of parallelogram . . . 84
16.4. The golden rectangle . . . 85
16.5. Regular polygon . . . 86

IV. Circles 88

17. Circles 89
17.1. Characteristics of a circle: \tkzDefCircle . . . 89
17.2. Projection of excenters . . . 93

18. Circles by transformation 96
18.1. The macro \tkzDefCircleBy . . . 96

V. Intersections and angles 99

19. Intersections 100
19.1. Intersection of two straight lines \tkzInterLL . . . 100
19.2. Intersection of a straight line and a circle \tkzInterLC . . . 100
19.3. Intersection of two circles \tkzInterCC . . . 106

20. Angles 110
20.1. Definition and usage with tkz-euclide . . . 110
20.2. Recovering an angle \tkzGetAngle . . . 111
20.3. Angle formed by three points . . . 111
20.4. Angle formed by a straight line with the horizontal axis \tkzFindSlopeAngle . . . 113

VI. Drawing, filling and clipping 115

21. Drawing 116
21.1. Draw a point or some points . . . 116

22. Drawing the lines 117
22.1. Draw a straight line . . . 117

23. Drawing a segment 118
23.1. Draw a segment \tkzDrawSegment . . . 118
23.2. Drawing segments \tkzDrawSegments . . . 121
23.3. Drawing the segments of a triangle . . . 122
23.4. Drawing a polygon . . . 122
23.5. Drawing a polygonal chain . . . 123

24. Draw a circle with \tkzDrawCircle 124
24.1. Draw one circle . . . 124
24.2. Drawing circles . . . 125
24.3. Drawing a semicircle . . . 128
24.4. Drawing semicircles . . . 128

25. Draw an ellipse with \tkzDrawEllipse 129
25.1. Draw an ellipse . . . 129

26. Drawing arcs 129
26.1. The macro \tkzDrawArc . . . 129

27. Drawing a sector or sectors 133
27.1. \tkzDrawSector . . . 133

28. Filling 135
28.1. Coloring a disc . . . 135
28.2. Coloring a polygon . . . 138
28.3. \tkzFillSector . . . 138
28.4. Coloring an angle: \tkzFillAngle . . . 139

29. Controlling the bounding box 141
29.1. Utility of \tkzInit . . . 141
29.2. \tkzInit . . . 141
29.3. \tkzClip . . . 142
29.4. \tkzClip and the option space . . . 142
29.5. \tkzShowBB . . . 143
29.6. \tkzClipBB . . . 143

30. Clipping different objects 145
30.1. Clipping a polygon . . . 145
30.2. Clipping a disc . . . 146
30.3. Clip out . . . 147
30.4. Intersection of disks . . . 147
30.5. Clipping a sector . . . 147
30.6. TikZ options: trim left or trim right . . . 148
30.7. TikZ Controls \pgfinterruptboundingbox and \endpgfinterruptboundingbox . . . 148
30.8. Reverse clip: tkzreverseclip . . . 149

VII. Marking and labelling 151

31. Marking 152
31.1. Mark a segment \tkzMarkSegment . . . 152
31.2. Marking segments \tkzMarkSegments . . . 152

31.3. Marking with a construction: equilateral triangle . . . 153
31.4. Mark an arc \tkzMarkArc . . . 153
31.5. Marking an angle: \tkzMarkAngle . . . 153
31.6. Marking a small angle: option veclen . . . 154
31.7. Marking a right angle: \tkzMarkRightAngle . . . 155
31.8. \tkzMarkRightAngles . . . 157
31.9. The TikZ library angles . . . 157

32. Labelling points 158
32.1. Label for a point: \tkzLabelPoint . . . 158
32.2. Add labels to points \tkzLabelPoints . . . 159
32.3. Automatic position of labels \tkzAutoLabelPoints . . . 159

33. Labelling segments 160
33.1. The macros \tkzLabelSegment and \tkzLabelSegments . . . 160

34. Labelling lines 162
34.1. The macro \tkzLabelLine . . . 162

35. Labelling angles 162
35.1. The macros \tkzLabelAngle and \tkzLabelAngles . . . 162

36. Labelling circles 164
36.1. The macro \tkzLabelCircle . . . 164

37. Labelling arcs 165
37.1. The macro \tkzLabelArc . . . 165

VIII. Complements 166

38. Using the compass 167
38.1. Main macro \tkzCompass . . . 167
38.2. Multiple constructions \tkzCompasss . . . 167

39. The Show 168
39.1. Show the constructions of some lines \tkzShowLine . . . 168
39.2. Constructions of certain transformations \tkzShowTransformation . . . 169

40. Protractor 171
40.1. The macro \tkzProtractor . . . 171

41. Miscellaneous and mathematical tools 172
41.1. Duplicate a segment . . . 172
41.2. Segment length \tkzCalcLength . . . 173
41.3. Transformation from pt to cm or cm to pt . . . 174
41.4. Change of unit . . . 175
41.5. Get point coordinates . . . 175
41.6. Swap labels of points . . . 176
41.7. Dot Product . . . 176
41.8. Power of a point with respect to a circle . . . 178
41.9. Radical axis . . . 178
41.10. \tkzIsLinear, \tkzIsOrtho . . . 180

IX. Working with styles 182

42. Predefined styles 183

43. Point style 183
43.1. Use of \tkzSetUpPoint . . . 183

44. Line style 185
44.1. Use of \tkzSetUpLine . . . 185

45. Arc style 187
45.1. The macro \tkzSetUpArc . . . 187

46. Compass style 188
46.1. The macro \tkzSetUpCompass . . . 188

47. Label style 188
47.1. The macro \tkzSetUpLabel . . . 188

48. Own style 189
48.1. The macro \tkzSetUpStyle . . . 189

49. How to use arrows 189
49.1. Arrows at endpoints on segment, ray or line . . . 190
49.2. Arrows on middle point of a line segment . . . 191
49.3. Arrows on all segments of a polygon . . . 192

X. Examples 194

50. Different authors 195
50.1. Code from Andrew Swan . . . 195
50.2. Example by Dimitris Kapetas . . . 195
50.3. Example by John Kitzmiller . . . 196
50.4. Example 1: from Indonesia . . . 197
50.5. Example 2: from Indonesia . . . 198
50.6. Illustration of the Morley theorem by Nicolas François . . . 200
50.7. Gougu theorem (Pythagorean theorem) by Zhao Shuang . . . 201
50.8. Reuleaux triangle . . . 202

51. Some interesting examples 203
51.1. Square root of the integers . . . 203
51.2. About right triangles . . . 204
51.3. Archimedes . . . 204
51.4. Lune of Hippocrates . . . 208
51.5. Lunes of Hasan Ibn al-Haytham . . . 208
51.6. About clipping circles . . . 210
51.7. Similar isosceles triangles . . . 211
51.8. Revised version of "Tangente" . . . 212
51.9. "Le Monde" version . . . 213
51.10. Triangle altitudes . . . 214
51.11. Altitudes - other construction . . . 215
51.12. Three circles in an equilateral triangle . . . 216
51.13. Law of sines . . . 217
51.14. Flower of Life . . . 218

51.15. Pentagon in a circle . . . 220
51.16. Pentagon in a square . . . 221
51.17. Inscribed hexagon . . . 222
51.18. Power of a point with respect to a circle . . . 223
51.19. Radical axis of two non-concentric circles . . . 224
51.20. External homothetic center . . . 225
51.21. Tangent lines to two circles . . . 226
51.22. Tangent lines to two circles with radical axis . . . 227
51.23. Midpoint of a segment with a compass . . . 229
51.24. Definition of a circle, _Apollonius_ . . . 230
51.25. Application of Inversion: **Pappus chain** . . . 231
51.26. Archimedes, Book of Lemmas, Proposition 1 . . . 232
51.27. Archimedes, Book of Lemmas, Proposition 6 . . . 232
51.28. "The" circle of Apollonius . . . 234

XI. FAQ 237

52. FAQ 238
52.1. Most common errors . . . 238

XII. News, compatibility and options 240

53. News and compatibility 241
53.1. With version 5.15 . . . 241
53.2. With version 5.14 . . . 241
53.3. With version 5.10 . . . 241
53.4. From version 3 to version 5 . . . 242

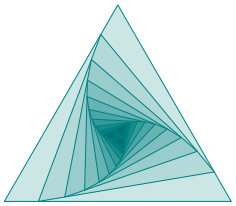
54. Working with lua 243
54.1. Option lua . . . 243
54.2. Option mini . . . 243

Index 244

Part I.

Getting started

1. Presentation and overview



```
\begin{tikzpicture}[scale=.25]
\tkzDefPoints{0/0/A,12/0/B,6/12*sind(60)/C}
\foreach \density in {20,30,...,240}{%
  \tkzDrawPolygon[fill=teal!\density](A,B,C)
  \pgfnodealias{X}{A}
  \tkzDefPointWith[linear,K=.15](A,B) \tkzGetPoint{A}
  \tkzDefPointWith[linear,K=.15](B,C) \tkzGetPoint{B}
  \tkzDefPointWith[linear,K=.15](C,X) \tkzGetPoint{C}}
\end{tikzpicture}
```

1.1. Why tkz-euclide?

My initial goal was to provide other mathematics teachers and myself with a tool to quickly create Euclidean geometry figures without investing too much effort in learning a new programming language. **tkz-euclide** is meant for mathematics teachers who use LaTeX: it makes it easy to create correct drawings within LaTeX.
The simplest method seemed to be to reproduce the way a construction is made by hand. To describe a construction, you must, of course, define the objects but also the actions that you perform. It seemed to me that a syntax close to the language of mathematicians and their students would be easier to understand, and that this syntax should also be close to that of LaTeX. The objects, of course, are points, segments, lines, triangles, polygons and circles. As for the actions, I considered five to be sufficient: define, create, draw, mark and label.
The syntax is perhaps verbose, but I believe it is easy to learn. As a result, students as well as teachers have been able to use this tool easily.

1.2. TikZ vs tkz-euclide

I love programming with TikZ, and without TikZ I would never have had the idea of creating **tkz-euclide**. Never forget that TikZ works behind it and that you can always insert TikZ code: **tkz-euclide** does not prevent you from using TikZ. That said, I don't think mixing syntax is a good thing.
There is no need to compare TikZ and **tkz-euclide**: they are not addressed to the same audience. The first one allows you to do a lot of things, the second one only draws geometric figures. The first one can do everything the second one does, but the second one will do what you want more easily.
The main purpose is to define points in order to create geometric figures. **tkz-euclide** allows you to draw the essential objects of Euclidean geometry from these points, but it may be insufficient for some actions such as coloring surfaces. In this case you can always use TikZ.
Here are some comparisons between **TikZ** and **tkz-euclide**, based on the geometry examples of the **PGF** manual. The two main tools used by the ancient Greeks to construct geometric shapes and angles were the compass and the straightedge. My idea is to let you follow, step by step and as naturally as possible, a construction that would be done by hand with these two tools.

1.2.1. Book I, Proposition I, _Euclid's Elements_

Book I, proposition I _Euclid's Elements_

To construct an equilateral triangle on a given finite straight line.

Explanation:
The fourth tutorial of the *PgfManual* is about geometric constructions. *T. Tantau* obtains the drawing with his beautiful tool TikZ. Here I propose the same construction with *tkz-euclide*. The TikZ code is shown in green and the *tkz-euclide* code in red.

```
\usepackage{tikz}
\usetikzlibrary{calc,intersections,through,backgrounds}
```

```
\usepackage{tkz-euclide}
```

How do we get the line AB? We use two fixed points.

```
\coordinate [label=left:$A$] (A) at (0,0);
\coordinate [label=right:$B$] (B) at (1.25,0.25);
\draw (A) -- (B);
```

```
\tkzDefPoint(0,0){A}
\tkzDefPoint(1.25,0.25){B}
\tkzDrawSegment(A,B)
\tkzLabelPoint[left](A){$A$}
\tkzLabelPoint[right](B){$B$}
```

We want to draw a circle around each of the points A and B, whose radius is the length of the segment AB.

```
\draw let \p1 = ($ (B) - (A) $),
\n2 = {veclen(\x1,\y1)} in
          (A) circle (\n2)
          (B) circle (\n2);
```

```
\tkzDrawCircles(A,B B,A)
```

The intersection of the circles $\mathscr{D}$ and $\mathscr{E}$:

```
draw [name path=A--B] (A) -- (B);
node (D) [name path=D,draw,circle through=(B),label=left:$D$] at (A) {};
node (E) [name path=E,draw,circle through=(A),label=right:$E$] at (B) {};
path [name intersections={of=D and E, by={[label=above:$C$]C,[label=below:$C'$]C'}}];
draw [name path=C--C',red] (C) -- (C');
path [name intersections={of=A--B and C--C',by=F}];
node [fill=red,inner sep=1pt,label=-45:$F$] at (F) {};
```

```
\tkzInterCC(A,B)(B,A) \tkzGetPoints{C}{X}
```

How to draw points:

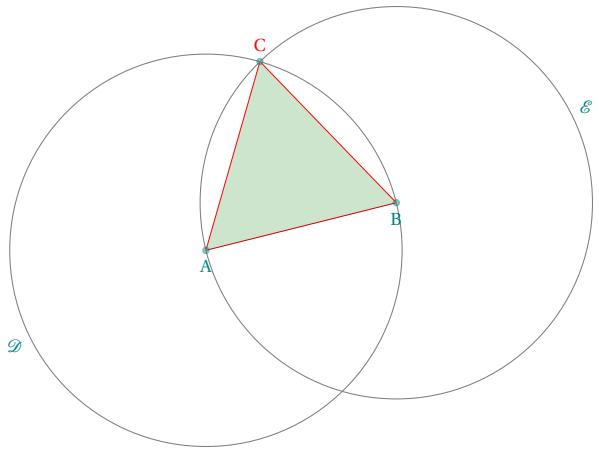
```
\foreach \point in {A,B,C}
\fill [black,opacity=.5] (\point) circle (2pt);
```

```
 \tkzDrawPoints[fill=gray,opacity=.5](A,B,C)
```

1.2.2. Complete code with tkz-euclide

We need to define some colors:

```
\colorlet{input}{red!80!black}
\colorlet{output}{red!70!black}
\colorlet{triangle}{green!50!black!40}
```




```
\colorlet{input}{red!80!black}
\colorlet{output}{red!70!black}
\colorlet{triangle}{green!50!black!40}
\begin{tikzpicture}[scale=1.25,thick,help lines/.style={thin,draw=black!50}]
\tkzDefPoint(0,0){A}
\tkzDefPoint(1.25+rand(),0.25+rand()){B}
\tkzInterCC(A,B)(B,A) \tkzGetPoints{C}{X}

\tkzFillPolygon[triangle,opacity=.5](A,B,C)
\tkzDrawSegment[input](A,B)
\tkzDrawSegments[red](A,C B,C)
\tkzDrawCircles[help lines](A,B B,A)
\tkzDrawPoints[fill=gray,opacity=.5](A,B,C)

\tkzLabelPoints(A,B)
\tkzLabelCircle[below=12pt](A,B)(180){$\mathcal{D}$}
\tkzLabelCircle[above=12pt](B,A)(180){$\mathcal{E}$}
\tkzLabelPoint[above,red](C){$C$}

\end{tikzpicture}
```

1.2.3. Book I, Proposition II, _Euclid's Elements_

Book I, Proposition II _Euclid's Elements_

To place a straight line equal to a given straight line with one end at a given point.

Explanation

In the first part, we need the midpoint of the segment AB. With Ti*k*Z we can use the library `calc`:

```
\coordinate [label=left:$A$] (A) at (0,0);
\coordinate [label=right:$B$] (B) at (1.25,0.25);
\draw (A) -- (B);
\node [fill=red,inner sep=1pt,label=below:$X$] (X) at ($ (A)!.5!(B) $) {};
```

With `tkz-euclide` we have the macro `\tkzDefMidPoint`; we get the point X with `\tkzGetPoint`, but we don't need this point for the next step.

```
\tkzDefPoints{0/0/A,0.75/0.25/B,1/1.5/C}
\tkzDefMidPoint(A,B) \tkzGetPoint{X}
```

Then we need to construct an equilateral triangle. It's easy with tkz-euclide. With TikZ it takes more effort, because you need the midpoint X to get the point D by a trigonometric calculation.

```
\node [fill=red,inner sep=1pt,label=below:$X$] (X) at ($ (A)!.5!(B) $) {};
\node [fill=red,inner sep=1pt,label=above:$D$] (D) at
($ (X) ! {sin(60)*2} ! 90:(B) $) {};
\draw (A) -- (D) -- (B);
```

```
\tkzDefTriangle[equilateral](A,B) \tkzGetPoint{D}
```

We can draw the triangle at the end of the picture with:

```
\tkzDrawPolygon{A,B,C}
```

We know how to draw the circle $\mathscr{H}$ around B through C and how to place the points E and F:

```
\node (H) [label=135:$H$,draw,circle through=(C)] at (B) {};
\draw (D) -- ($ (D) ! 3.5 ! (B) $) coordinate [label=below:$F$] (F);
\draw (D) -- ($ (D) ! 2.5 ! (A) $) coordinate [label=below:$E$] (E);
```

```
\tkzDrawCircle(B,C)
\tkzDrawLines[add=0 and 2](D,A D,B)
```

We can place the points E and F at the end of the picture. We don't need them now.
Intersecting a line and a circle: here we look for the intersection of the circle around B through C with the line DB. The infinite line DB intersects the circle, but with Ti*k*Z we need to extend the segment DB, which can be done with partway calculations. We get the point F, and BF or DF intersects the circle:

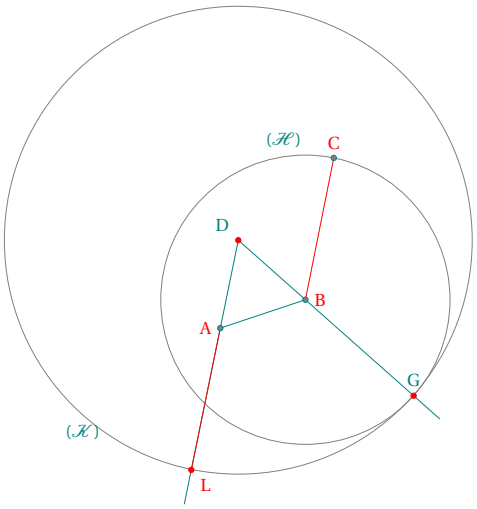
```
 \node (H) [label=135:$H$,draw,circle through=(C)] at (B) {};
\path let \p1 = ($ (B) - (C) $) in
  coordinate [label=left:$G$] (G) at ($ (B) ! veclen(\x1,\y1) ! (F) $);
\fill[red,opacity=.5] (G) circle (2pt);
```

As for the intersection of two circles, it's easy to find the intersection of a line and a circle with tkz-euclide. We don't need F:

```
 \tkzInterLC(B,D)(B,C)\tkzGetFirstPoint{G}
```

There are no more difficulties. Here is the final code, with some simplifications. We draw the circle $\mathscr{K}$ with center D and passing through G. It intersects the line AD at point L. AL = BC.

```
 \tkzDrawCircle(D,G)
 \tkzInterLC(D,A)(D,G)\tkzGetSecondPoint{L}
```




```
\begin{tikzpicture}[scale=1.5]
\tkzDefPoint(0,0){A}
\tkzDefPoint(0.75,0.25){B}
\tkzDefPoint(1,1.5){C}
\tkzDefTriangle[equilateral](A,B)\tkzGetPoint{D}
\tkzInterLC[near](D,B)(B,C)       \tkzGetSecondPoint{G}
\tkzInterLC[near](A,D)(D,G)       \tkzGetFirstPoint{L}
\tkzDrawCircles(B,C D,G)
\tkzDrawLines[add=0 and 2](D,A D,B)
\tkzDrawSegment(A,B)
\tkzDrawSegments[red](A,L B,C)
\tkzDrawPoints[red](D,L,G)
\tkzDrawPoints[fill=gray](A,B,C)
\tkzLabelPoints[left,red](A)
\tkzLabelPoints[below right,red](L)
\tkzLabelCircle[above](B,C)(20){$\mathcal{(H)}$}
\tkzLabelPoints[above left](D)
\tkzLabelPoints[above](G)
\tkzLabelPoints[above,red](C)
\tkzLabelPoints[right,red](B)
\tkzLabelCircle[below](D,G)(-90){$\mathcal{(K)}$}
\end{tikzpicture}
```

1.3. How to use the tkz-euclide package?

1.3.1. Let's look at a classic example

To show the right way, we will build an equilateral triangle. Several methods are possible; we will follow the steps of Euclid.

– First of all, you need a document class. To test your code, the best choice is to create a single figure with the class **standalone**.

```
\documentclass{standalone}
```

– Then load the **tkz-euclide** package:

```
\usepackage{tkz-euclide} or \usepackage[lua]{tkz-euclide}
```

You don't need to load Ti*k*Z: the **tkz-euclide** package works on top of TikZ and loads it.

– Start the document and open a TikZ picture environment:

```
\begin{document}
\begin{tikzpicture}
```

– Now we define two fixed points:

```
\tkzDefPoint(0,0){A}
\tkzDefPoint(5,2){B}
```

– Two points define two circles; let's use them:

circle with center A through B and circle with center B through A. These two circles have two points in common.

```
\tkzInterCC(A,B)(B,A)
```

We get the points of intersection with

```
\tkzGetPoints{C}{D}
```

– All the necessary points are now defined; we can move on to the final steps, including the drawings.

```
\tkzDrawCircles[gray,dashed](A,B B,A)
\tkzDrawPolygon(A,B,C)% The triangle
```

– Draw all points A, B, C and D:

```
\tkzDrawPoints(A,...,D)
```

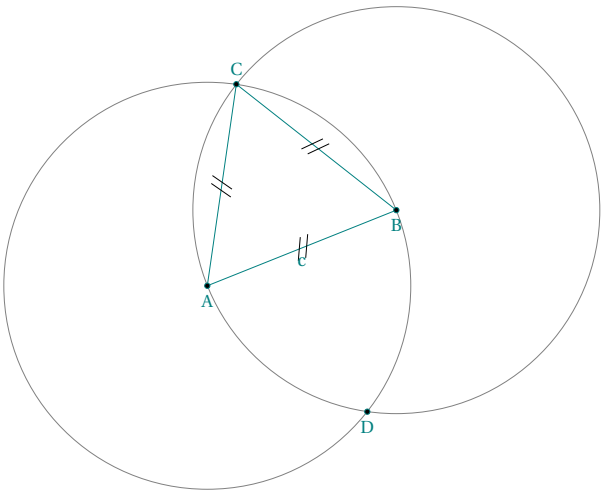
– In the final step, we add labels to the points and use options to position them:

```
\tkzLabelSegments[swap](A,B){$c$}
\tkzLabelPoints(A,B,D)
\tkzLabelPoints[above](C)
```

– Finally, we close both environments:

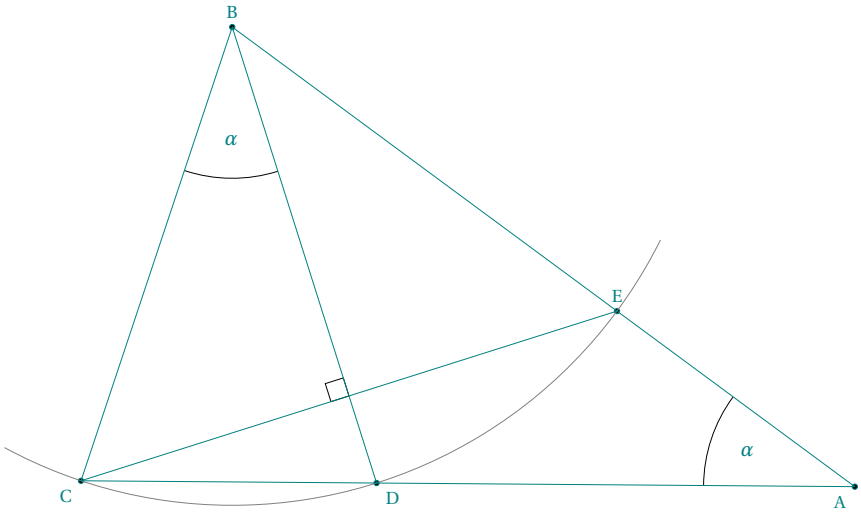
```
\end{tikzpicture}
\end{document}
```

– The complete code:




```
\begin{tikzpicture}[scale=.5]
  % fixed points
 \tkzDefPoint(0,0){A}
 \tkzDefPoint(5,2){B}
 % calculus
 \tkzInterCC(A,B)(B,A)
 \tkzGetPoints{C}{D}
 % drawings
 \tkzDrawCircles(A,B B,A)
 \tkzDrawPolygon(A,B,C)
 \tkzDrawPoints(A,...,D)
 % marking
 \tkzMarkSegments[mark=s||](A,B B,C C,A)
 % labelling
 \tkzLabelSegments[swap](A,B){$c$}
 \tkzLabelPoints(A,B,D)
 \tkzLabelPoints[above](C)
\end{tikzpicture}
```

1.3.2. Part I: golden triangle



Let's analyze the figure:

1. CBD and DBE are isosceles triangles;
2. BC = BE and (BD) is a bisector of the angle CBE;
3. From this we deduce that the angles CBD and DBE are equal; let α be their measure.

$$\widehat{BAC} + \widehat{ABC} + \widehat{BCA} = 180^\circ \text{ in the triangle BAC}$$

$$3\alpha + \widehat{BCA} = 180^\circ \text{ in the triangle CBD}$$

then

$$\alpha + 2\widehat{BCA} = 180^\circ$$

or

$$\widehat{BCA} = 90^\circ - \alpha/2$$

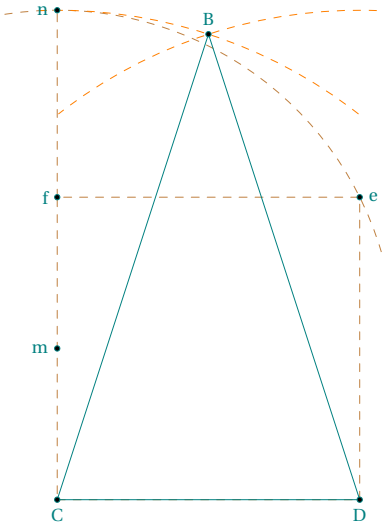
4. Finally

$$\widehat{CBD} = \alpha = 36^\circ$$

the triangle CBD is a `golden` triangle.

How can we construct a golden triangle, or an angle of 36°?

1. We place the fixed points C and D. `\tkzDefPoint(0,0){C}` and `\tkzDefPoint(4,0){D}`;
2. We construct a square CDef and we construct the midpoint m of [Cf];
   We can do all of this with a compass and a ruler;
3. Then we draw an arc with center m through e. This arc crosses the line (Cf) at n;
4. Now the two arcs with center C and D and radius Cn define the point B.




```
\begin{tikzpicture}
  \tkzDefPoint(0,0){C}
  \tkzDefPoint(4,0){D}
  \tkzDefSquare(C,D)
  \tkzGetPoints{e}{f}
  \tkzDefMidPoint(C,f)
  \tkzGetPoint{m}
  \tkzInterLC(C,f)(m,e)
  \tkzGetSecondPoint{n}
  \tkzInterCC[with nodes](C,C,n)(D,C,n)
  \tkzGetFirstPoint{B}
  \tkzDrawSegment[brown,dashed](f,n)
  \pgfinterruptboundingbox% from tikz
  \tkzDrawPolygon[brown,dashed](C,D,e,f)
  \tkzDrawArc[brown,dashed](m,e)(n)
  \tkzCompass[brown,dashed,delta=20](C,B)
  \tkzCompass[brown,dashed,delta=20](D,B)
  \endpgfinterruptboundingbox
  \tkzDrawPolygon(B,...,D)
  \tkzDrawPoints(B,C,D,e,f,m,n)
  \tkzLabelPoints[above](B)
  \tkzLabelPoints[left](f,m,n)
  \tkzLabelPoints(C,D)
  \tkzLabelPoints[right](e)
\end{tikzpicture}
```

After building the golden triangle BCD, we build the point A by noticing that BD = DA. Then we get the point E and finally the point F, as intersections of objects already defined (a line and a circle).

1.3.3. Part II: two other methods with the golden and Euclid triangles

tkz-euclide knows how to define a golden or euclide triangle. We can define BCD and BCA as golden triangles.

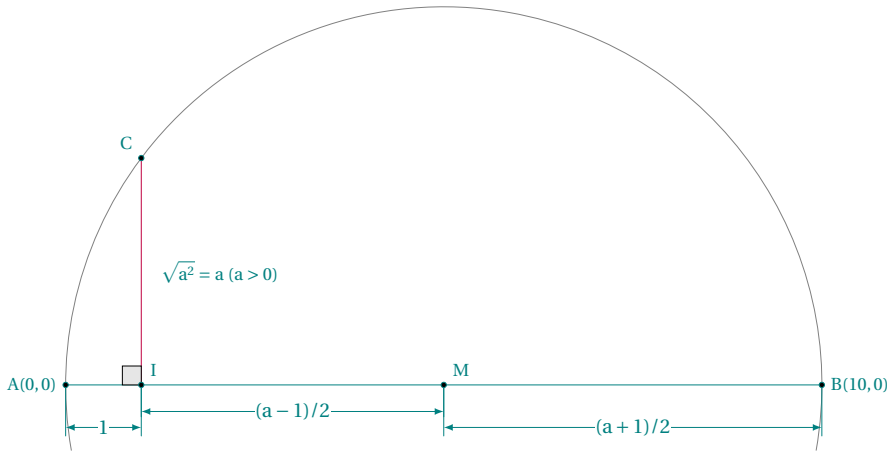
```
\begin{tikzpicture}
  \tkzDefPoint(0,0){C}
  \tkzDefPoint(4,0){D}
  \tkzDefTriangle[golden](C,D)
  \tkzGetPoint{B}
  \tkzDefTriangle[golden](B,C)
  \tkzGetPoint{A}
  \tkzInterLC[near](A,B)(B,D) \tkzGetFirstPoint{E}
  \tkzInterLL(B,D)(C,E) \tkzGetPoint{F}
  \tkzDrawPoints(C,D,B)
  \tkzDrawPolygon(B,...,D)
  \tkzDrawPolygon(B,C,D)
  \tkzDrawSegments(D,A A,B C,E)
  \tkzDrawArc[delta=10](B,C)(E)
  \tkzDrawPoints(A,...,F)
  \tkzMarkRightAngle(B,F,C)
  \tkzMarkAngles(C,B,D E,A,D)
  \tkzLabelAngles[pos=1.5](C,B,D E,A,D){$\alpha$}
  \tkzLabelPoints[below](A,C,D,E)
  \tkzLabelPoints[above right](B,F)
\end{tikzpicture}
```

Here is a final method that uses rotations:

```
\begin{tikzpicture}
\tkzDefPoint(0,0){C} % possible
% \tkzDefPoint[label=below:$C$](0,0){C}
% but don't do this
\tkzDefPoint(2,6){B}
% We get D and E with a rotation
\tkzDefPointBy[rotation= center B angle 36](C) \tkzGetPoint{D}
\tkzDefPointBy[rotation= center B angle 72](C) \tkzGetPoint{E}
% To get A we use an intersection of lines
\tkzInterLL(B,E)(C,D) \tkzGetPoint{A}
\tkzInterLL(C,E)(B,D) \tkzGetPoint{H}
% drawing
\tkzDrawArc[delta=10](B,C)(E)
\tkzDrawPolygon(C,B,D)
\tkzDrawSegments(D,A B,A C,E)
% angles
\tkzMarkAngles(C,B,D E,A,D) %this is to draw the arcs
\tkzLabelAngles[pos=1.5](C,B,D E,A,D){$\alpha$}
\tkzMarkRightAngle(B,H,C)
\tkzDrawPoints(A,...,E)
% Label only now
\tkzLabelPoints[below left](C,A)
\tkzLabelPoints[below right](D)
\tkzLabelPoints[above](B,E)
\end{tikzpicture}
```

1.3.4. Complete but minimal example

A unit of length being chosen, this example shows how to obtain a segment of length $\sqrt{a}$ from a segment of length a, with a ruler and a compass.
IB = a, AI = 1



```
\begin{tikzpicture}[scale=1,ra/.style={fill=gray!20}]
   % fixed points
   \tkzDefPoint(0,0){A}
   \tkzDefPoint(1,0){I}
   % calculation
   \tkzDefPointBy[homothety=center A ratio  10 ](I) \tkzGetPoint{B}
   \tkzDefMidPoint(A,B)              \tkzGetPoint{M}
   \tkzDefPointWith[orthogonal](I,M) \tkzGetPoint{H}
   \tkzInterLC(I,H)(M,B)             \tkzGetFirstPoint{C}
   \tkzDrawSegment[style=purple](I,C)
   \tkzDrawArc(M,B)(A)
   \tkzDrawSegment[dim={$1$,-16pt,}](A,I)
   \tkzDrawSegment[dim={$(a-1)/2$,-10pt,}](I,M)
   \tkzDrawSegment[dim={$(a+1)/2$,-16pt,}](M,B)
   \tkzMarkRightAngle[ra](A,I,C)
   \tkzDrawPoints(I,A,B,C,M)
   \tkzLabelPoint[left](A){$A(0,0)$}
   \tkzLabelPoints[above right](I,M)
   \tkzLabelPoints[above left](C)
   \tkzLabelPoint[right](B){$B(10,0)$}
   \tkzLabelSegment[right=4pt](I,C){$\sqrt{a^2}=a \ (a>0)$}
\end{tikzpicture}
```

Comments

– The Preamble

Let us first look at the preamble. If you need **xcolor**, load it before **tkz-euclide**, that is, before Ti*k*Z. Ti*k*Z may cause problems with active characters, but it provides the library babel, which is meant to solve these problems.

```
\documentclass{standalone} % or another class
   % \usepackage{xcolor} % before tikz or tkz-euclide if necessary
\usepackage{tkz-euclide} % no need to load TikZ
   % \usetkzobj{all}  is no longer necessary
   % \usetikzlibrary{babel}  if there are problems with the active characters
```

The following code consists of several parts:

– Definition of the fixed points: the first part defines the points needed for the construction; these are the fixed points. In most cases the macros **\tkzInit** and **\tkzClip** are not necessary.

```
\tkzDefPoint(0,0){A}
\tkzDefPoint(1,0){I}
```

– The second part creates new points from the fixed points: a point B is placed 10 cm from A. The midpoint of [AB] is M, then we define the line orthogonal to (AB) at the point I. Then we look for the intersection of this line with the semicircle with center M through A.

```
\tkzDefPointBy[homothety=center A ratio  10 ](I)
   \tkzGetPoint{B}
\tkzDefMidPoint(A,B)
   \tkzGetPoint{M}
\tkzDefPointWith[orthogonal](I,M)
   \tkzGetPoint{H}
\tkzInterLC(I,H)(M,B)
```

```
\tkzGetSecondPoint{C}
```

– The third part contains the drawings;

```
\tkzDrawSegment[style=purple](I,H)
\tkzDrawPoints(O,I,A,B,M)
\tkzDrawArc(M,A)(O)
\tkzDrawSegment[dim={$1$,-16pt,}](A,I)
\tkzDrawSegment[dim={$a/2$,-10pt,}](I,M)
\tkzDrawSegment[dim={$a/2$,-16pt,}](M,B)
```

– Marking: the fourth part is devoted to marks;

```
\tkzMarkRightAngle[ra](A,I,C)
```

– Labelling: the last part only places the labels.

```
\tkzLabelPoint[left](A){$A(0,0)$}
\tkzLabelPoint[right](B){$B(10,0)$}
\tkzLabelSegment[right=4pt](I,C){$\sqrt{a^2}=a \ (a>0)$}
```

2. Installation

tkz-euclide is available on **CTAN**[1]. If you want to test a beta version, put the files in a texmf folder that your system can find. Check the following points:

– The **tkz-euclide** folder must be in a path searched by **latex**.

– **tkz-euclide** uses the package xfp.

– **PGF** must be installed on your computer. **tkz-euclide** uses several libraries of Ti*k*Z:
angles,
arrows,
arrows.meta,
backgrounds,
calc,
decorations,
decorations.markings,
decorations.pathreplacing,
decorations.shapes,
decorations.text,
decorations.pathmorphing,
intersections,
math,
plotmarks,
positioning,
quotes,
shapes.misc,
through

– This documentation was compiled with **lualatex**, but **pdflatex** and **xelatex** can also be used.

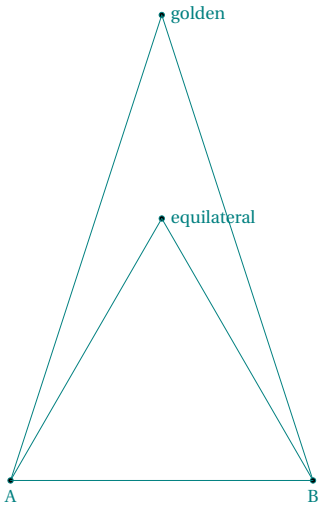
1 **tkz-euclide** is part of TeXLive, where **tlmgr** installs it, and of MiKTeX under Windows.

3. The Elements of tkz code

To work with this package, you need some knowledge of LaTeX and of Ti*k*Z.
In this section, we look at the `rules` and `symbols` used to create a figure with **tkz-euclide**.

3.1. Objects and language

The primitive objects are points. You can refer to a point at any time with the name given when it was defined (it is possible to give it another name later).
New points are obtained with macros. The names of the **tkz-euclide** macros begin with `tkz`. There are four main categories, starting with: `\tkzDef...` `\tkzDraw...` `\tkzMark...` and `\tkzLabel...`. The points used are passed as arguments between parentheses, while the points created are given between braces.
The code of a figure is placed in a `tikzpicture` environment.
Contrary to Ti*k*Z, a macro must not end with ";". We thus lose the important notion of `path`. However, you can place some Ti*k*Z code between the **tkz-euclide** macros.
In the first category, `\tkzDefPoint` defines fixed points; it will be studied in detail later. Here we look at the macro `\tkzDefTriangle`.
This macro associates a third point with a pair of points, in order to define a certain triangle: `\tkzDefTriangle(A,B)`.
The point obtained is referenced `tkzPointResult`, and you can choose another reference, for example with `\tkzGetPoint{C}`.
`\tkzDefTriangle[euclid](A,B) \tkzGetPoint{C}`
Parentheses are used to pass arguments: in `(A,B)`, A and B are the points from which a third one will be defined.
In `{C}`, braces are used to retrieve the new point.
To choose a type of triangle among the following: `equilateral`, `isosceles right`, `half`, `pythagoras`, `school`, `golden or sublime`, `euclid`, `gold`, `cheops`... and `two angles`, you just give the option between square brackets, for example:




```
\begin{tikzpicture}[scale=.5]
\tkzDefPoints{0/0/A,8/0/B}
\foreach \tr in {golden, equilateral}
{\tkzDefTriangle[\tr](A,B) \tkzGetPoint{C}
\tkzDrawPoint(C)
\tkzLabelPoint[right](C){\tr}
\tkzDrawSegments(A,C C,B)}
\tkzDrawPoints(A,B)
\tkzDrawSegments(A,B)
\tkzLabelPoints(A,B)
\end{tikzpicture}
```

3.2. Notations and conventions

I deliberately chose French geometric conventions, and some personal ones, to describe the objects represented.
The objects defined and drawn by **tkz-euclide** are points, lines and circles in a plane. They are the primary objects of Euclidean geometry, from which we construct figures.
According to `Euclid`, these figures only illustrate pure ideas produced by our mind. Thus a point has no dimension and therefore no real existence. In the same way, a line has no width and therefore no existence in the real world. The objects we consider are only representations of ideal mathematical objects. **tkz-euclide** follows the steps of the ancient Greeks to obtain geometric constructions with ruler and compass.
Here are the notations used:

– Points are represented either by a small disc or by the intersection of two lines (two straight lines, a straight line and a circle, or two circles). In the second case, the point is represented by a cross.




```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,4/2/B}
  \tkzDrawPoints(A,B)
  \tkzLabelPoints(A,B)
\end{tikzpicture}
```

or else




```
\begin{tikzpicture}
  \tkzSetUpPoint[shape=cross, color=red]
  \tkzDefPoints{0/0/A,4/2/B}
  \tkzDrawPoints(A,B)
  \tkzLabelPoints(A,B)
  \end{tikzpicture}
```

Once a point exists, we can give it a label, which is a capital letter of the Latin alphabet (with some exceptions) such as A, B or C. For example:

  – O is a center for a circle, a rotation, etc.;
  – M is a midpoint;
  – H is the foot of an altitude;
  – P′ is the image of P by a transformation.

Note that the reference name of a point in the code may differ from the label that designates it in the text: we can define a point A and give it the label P. The style is also different: the point A is labelled A.




```
\begin{tikzpicture}
  \tkzDefPoint(0,0){A}
  \tkzDrawPoints(A)
  \tkzLabelPoint(A){$P$}
\end{tikzpicture}
```

Exceptions: some points, such as the midpoints of the sides of a triangle, share a property, so their names also share a common letter. We designate these points by M_a, M_b and M_c, or M_A, M_B and M_C.

In the code, these points are referred to as M_A, M_B and M_C.

Another exception concerns intermediate construction points, which are not labelled. They are often designated by a lowercase letter in the code.

– Segments are designated by their two endpoints in square brackets: [AB].

– In Euclidean geometry, a straight line is defined by two points: A and B define the line (AB). We can also name a line with a Greek letter, (δ) or (Δ), or with a lowercase letter such as d or d′.

– A ray (half-line) is written [AB).

– Relations between lines: if (AB) and (CD) are perpendicular we write (AB) $\perp$ (CD), and if they are parallel we write (AB) $\parallel$ (CD).

– The lengths of the sides of the triangle ABC are AB, AC and BC. Numbers are also designated by lowercase letters, so we write AB = c, AC = b and BC = a. The letter a is also used for an angle; r is often used for a radius, d for a diameter or a distance, l for a length.

– Polygons are designated by their vertices: ABC is a triangle, EFGH a quadrilateral.

- Angles are generally measured in degrees (for example 60°); in an equilateral triangle ABC we write $\widehat{ABC} = \widehat{B} = 60°$.
- Arcs are designated by their endpoints: if A and B are two points of a circle, we write $\overset{\frown}{AB}$.
- A circle is written $\mathscr{C}$ if there is no possible confusion, $\mathscr{C}$ (O ; A) for the circle with center O through the point A, or $\mathscr{C}$ (O ; 1) for the circle with center O and radius 1 cm.
- Names of the special lines of a triangle: I use the terms bisector, bisector out (external bisector), mediator (usually called perpendicular bisector), altitude, median and symmedian.
- (x_1,y_1) are the coordinates of the point A_1, (x_A,y_A) those of the point A.

3.3. Set, Calculate, Draw, Mark, Label

This section could also have been called `Separation of calculations and drawings`.
When a document is prepared with LaTeX, its source code is divided into two parts: the preamble and the body of the document. With this method, documents can be structured, styled and typeset with minimal effort. I propose a similar method to create figures with **tkz-euclide**.
The first part defines the fixed points, the second one creates new points: **Set and Calculate** are the two main parts. Then all that is left to do is to draw (or fill), mark and label. **tkz-euclide** may be insufficient for some of these last actions, but you can always use Ti*k*Z.
One last remark, which I think is important: avoid coordinates in the code as much as possible. Coordinates should only appear at the beginning, with the fixed points; after that, use the references of the points. Most macros have the option **nodes** or **with nodes**.
It is also best to define the styles of the different objects at the beginning.

4. About this documentation and the examples

This documentation is compiled with `lualatex`. I use my own class **tkz-doc.cls**, based on **scrartcl**.
Here is the list of the styles used in this documentation. To understand how to use styles, see section 42.

```
\tkzSetUpColors[background=white,text=black]
\tkzSetUpCompass[color=orange, line width=.2pt,delta=10]
\tkzSetUpArc[color=gray,line width=.2pt,delta=10]
\tkzSetUpPoint[size=2,color=teal]
\tkzSetUpLine[line width=.2pt,color=teal]
\tkzSetUpStyle[color=orange,line width=.2pt]{new}
\tikzset{every picture/.style={line width=.2pt}}
\tikzset{label angle style/.append style={color=teal,font=\footnotesize}}
\tikzset{label style/.append style={below,color=teal,font=\scriptsize}}
```

Some examples use predefined styles such as:

```
\tikzset{new/.style={color=orange,line width=.2pt}}
```

Part II.

Points

5. First step: fixed points

The first step in a geometric construction is to define the fixed points from which the figure will be constructed. The general idea is to avoid manipulating coordinates, and to use instead the references of the points fixed in the first step or obtained with the tools of the package. Even if it is possible, I think it's a bad idea to work directly with coordinates: it is better to use named points.
tkz-euclide uses macros and a vocabulary specific to geometric construction. You can of course use the tools of Ti*k*Z, but it seems more logical to me not to mix the two syntaxes.
For Ti*k*Z, a point of **tkz-euclide** is a particular `node`. In the next section we will see how to define points with coordinates. The style of the points (color and shape) is not discussed here: you will find some indications in the examples, and more information in section 42.

6. Definition of a point: \tkzDefPoint or \tkzDefPoints

Points can be specified in any of the following ways:

- Cartesian coordinates;
- Polar coordinates;
- Named points;
- Relative points.

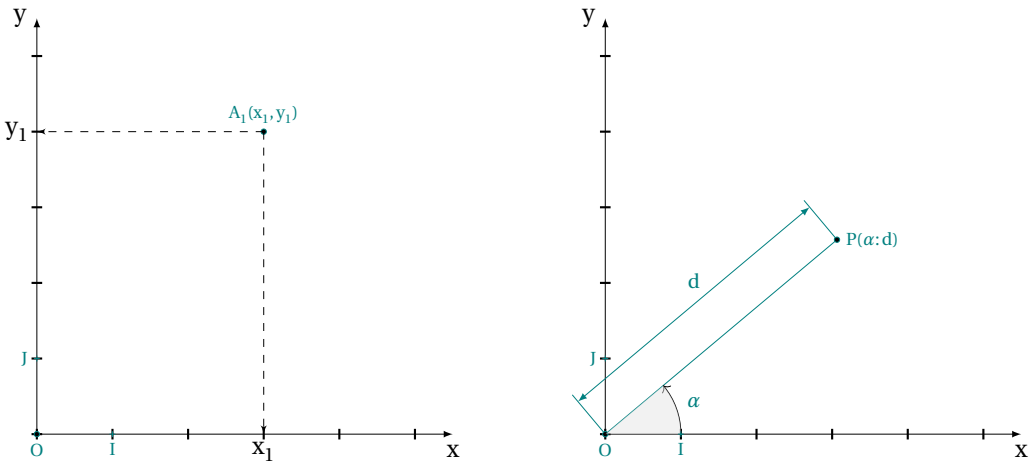
A point is defined when its name is linked to a unique pair of decimal numbers: (x,y) (x abscissa, y ordinate) or (a:d) (a angle, d distance). This is possible because the plane is given an orthonormal Cartesian coordinate system, whose unit is 1 cm.
The Cartesian coordinate (a,b) refers to the point a centimeters in the x-direction and b centimeters in the y-direction.
A point in polar coordinates requires an angle α, in degrees, and a distance d from the origin; the default unit is the `cm`.
The macro **\tkzDefPoint** defines a point by assigning coordinates to it. This macro is based on **\coordinate**, a macro of Ti*k*Z, and accepts Ti*k*Z options such as **shift**. Calculations are done with the package `xfp`. We can use Cartesian or polar coordinates.

Cartesian coordinates

```
\begin{tikzpicture}[scale=1]
  \tkzInit[xmax=5,ymax=5]
  % necessary to limit
  % the size of the axes
  \tkzDrawX[>=latex]
  \tkzDrawY[>=latex]
  \tkzDefPoints{0/0/O,1/0/I,0/1/J}
  \tkzDefPoint(3,4){A}
  \tkzDrawPoints(O,A)
  \tkzLabelPoint[above](A){$A_1(x_1,y_1)$}
  \tkzShowPointCoord[xlabel=$x_1$,
                     ylabel=$y_1$](A)
  \tkzLabelPoints(O,I)
  \tkzLabelPoints[left](J)
  \tkzDrawPoints[shape=cross](I,J)
\end{tikzpicture}
```

Polar coordinates

```
\begin{tikzpicture}[scale=1]
  \tkzInit[xmax=5,ymax=5]
  \tkzDrawX[>=latex]
  \tkzDrawY[>=latex]
  \tkzDefPoints{0/0/O,1/0/I,0/1/J}
  \tkzDefPoint(40:4){P}
  \tkzDrawSegment[dim={$d$,
                 16pt,above=6pt}](O,P)
  \tkzDrawPoints(O,P)
  \tkzMarkAngle[mark=none,->](I,O,P)
  \tkzFillAngle[opacity=.5](I,O,P)
  \tkzLabelAngle[pos=1.25](I,O,P){%
                              $\alpha$}
  \tkzLabelPoint[right](P){$P(\alpha:d)$}
  \tkzDrawPoints[shape=cross](I,J)
  \tkzLabelPoints(O,I)
  \tkzLabelPoints[left](J)
\end{tikzpicture}
```



6.1. Defining a named point: \tkzDefPoint

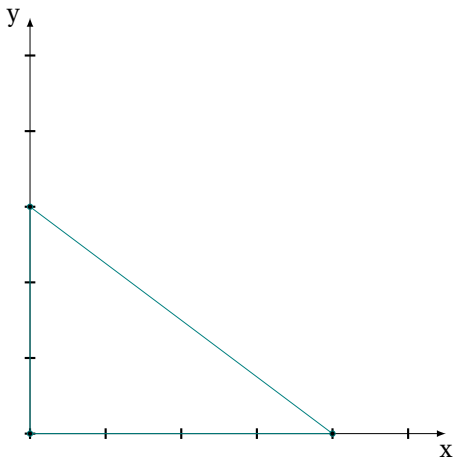
\tkzDefPoint[⟨local options⟩](⟨x,y⟩){⟨ref⟩} or (⟨α:d⟩){⟨ref⟩}

arguments	default	definition
(x,y)	no default	x and y are two dimensions, by default in cm.
(α:d)	no default	α is an angle in degrees, d is a dimension
{ref}	no default	Reference assigned to the point: A, T_a ,P1 or P_1

The mandatory arguments of this macro are two decimal numbers: in the first case two lengths, in the second case an angle in degrees and a length. Do not confuse the reference of a point with its name: the reference is used by the calculations, but the name is frequently identical to the reference.

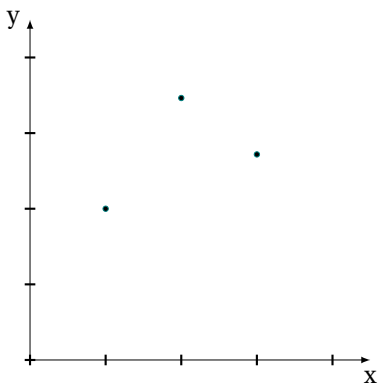
options	default	definition
label	no default	allows you to place a label at a predefined distance
shift	no default	adds (x,y) or (α:d) to all coordinates

6.1.1. Cartesian coordinates



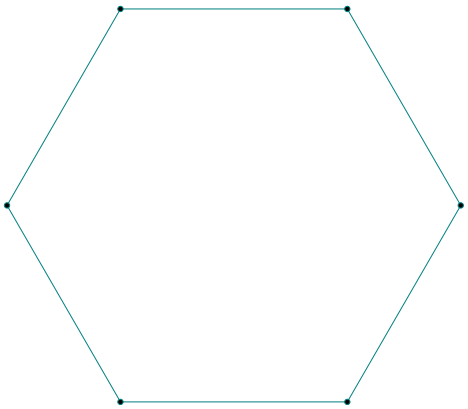

```
\begin{tikzpicture}
\tkzInit[xmax=5,ymax=5] % limits the size of the axes
\tkzDrawX[>=latex]
\tkzDrawY[>=latex]
\tkzDefPoint(0,0){A}
\tkzDefPoint(4,0){B}
\tkzDefPoint(0,3){C}
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints(A,B,C)
\end{tikzpicture}
```

6.1.2. Calculations with xfp




```
\begin{tikzpicture}[scale=1]
  \tkzInit[xmax=4,ymax=4]
  \tkzDrawX\tkzDrawY
  \tkzDefPoint(-1+2,sqrt(4)){O}
  \tkzDefPoint({3*ln(exp(1))},{exp(1)}){A}
  \tkzDefPoint({4*sin(pi/6)},{4*cos(pi/6)}){B}
  \tkzDrawPoints(O,B,A)
\end{tikzpicture}
```

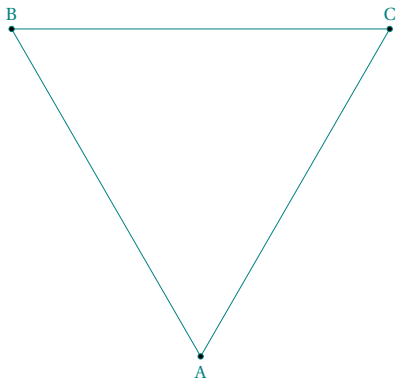
6.1.3. Polar coordinates



```
  \begin{tikzpicture}
  \foreach \an [count=\i] in {0,60,...,300}
   { \tkzDefPoint(\an:3){A_\i}}
  \tkzDrawPolygon(A_1,A_...,A_6)
  \tkzDrawPoints(A_1,A_...,A_6)
  \end{tikzpicture}
```

6.1.4. Relative points

First, we can use the **scope** environment from Ti*k*Z. In the following example, we have a way to define an equilateral triangle.




```
\begin{tikzpicture}[scale=1]
 \begin{scope}[rotate=30]
  \tkzDefPoint(2,3){A}
  \begin{scope}[shift=(A)]
     \tkzDefPoint(90:5){B}
     \tkzDefPoint(30:5){C}
  \end{scope}
 \end{scope}
 \tkzDrawPolygon(A,B,C)
\tkzLabelPoints[above](B,C)
\tkzLabelPoints[below](A)
\tkzDrawPoints(A,B,C)
\end{tikzpicture}
```

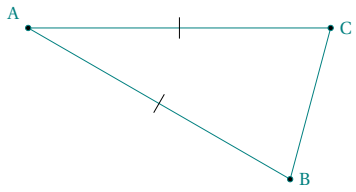
6.2. Point relative to another: \tkzDefShiftPoint

\tkzDefShiftPoint[⟨Point⟩](⟨x,y⟩){⟨ref⟩} or (⟨α:d⟩){⟨ref⟩}

arguments	default	definition
(x,y)	no default	x and y are two dimensions, by default in cm.
(α:d)	no default	α is an angle in degrees, d is a dimension
{ref}	no default	Reference assigned to the point: A, T_a ,P1 or P_1
options	default	definition
[pt]	no default	\tkzDefShiftPoint[A](0:4){B}

6.2.1. Isosceles triangle

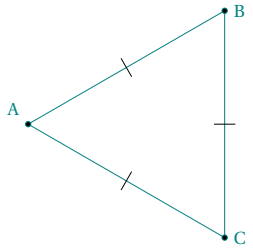
This macro places a point relative to another one; it is equivalent to a translation. Here is how to construct an isosceles triangle with main vertex A and an angle of 30° at this vertex.




```
\begin{tikzpicture}[rotate=-30]
 \tkzDefPoint(2,3){A}
 \tkzDefShiftPoint[A](0:4){B}
 \tkzDefShiftPoint[A](30:4){C}
 \tkzDrawSegments(A,B B,C C,A)
 \tkzMarkSegments[mark=|](A,B A,C)
 \tkzDrawPoints(A,B,C)
 \tkzLabelPoints[right](B,C)
 \tkzLabelPoints[above left](A)
\end{tikzpicture}
```

6.2.2. Equilateral triangle

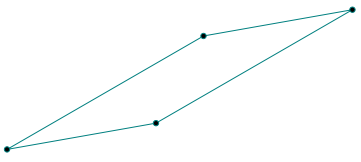
Let's see how to get an equilateral triangle (there is a much simpler way, see \tkzDefTriangle).




```
\begin{tikzpicture}[scale=1]
 \tkzDefPoint(2,3){A}
 \tkzDefShiftPoint[A](30:3){B}
 \tkzDefShiftPoint[A](-30:3){C}
 \tkzDrawPolygon(A,B,C)
 \tkzDrawPoints(A,B,C)
 \tkzLabelPoints[right](B,C)
 \tkzLabelPoints[above left](A)
 \tkzMarkSegments[mark=|](A,B A,C B,C)
\end{tikzpicture}
```

6.2.3. Parallelogram

There is a simpler way, see \tkzDefParallelogram.



```
\begin{tikzpicture}
 \tkzDefPoint(0,0){A}
 \tkzDefPoint(30:3){B}
 \tkzDefShiftPointCoord[B](10:2){C}
 \tkzDefShiftPointCoord[A](10:2){D}
 \tkzDrawPolygon(A,...,D)
 \tkzDrawPoints(A,...,D)
\end{tikzpicture}
```

6.3. Definition of multiple points: \tkzDefPoints

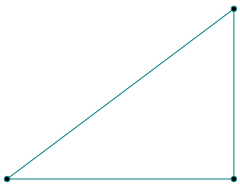
\tkzDefPoints[⟨local options⟩]{⟨x_1/y_1/r_1,x_2/y_2/r_2, ...⟩}

x_i and y_i are the coordinates of the point with reference r_i.

arguments	default	example
x_i/y_i/r_i		\tkzDefPoints{0/0/O,2/2/A}

options	default	definition
shift	no default	Adds (x,y) or (α:d) to all coordinates

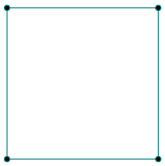
6.4. Create a triangle



```
\begin{tikzpicture}[scale=.75]
 \tkzDefPoints{0/0/A,4/0/B,4/3/C}
 \tkzDrawPolygon(A,B,C)
 \tkzDrawPoints(A,B,C)
\end{tikzpicture}
```

6.5. Create a square

Note here the syntax for drawing the polygon.



```
\begin{tikzpicture}[scale=1]
 \tkzDefPoints{0/0/A,2/0/B,2/2/C,0/2/D}
 \tkzDrawPolygon(A,...,D)
 \tkzDrawPoints(A,...,D)
\end{tikzpicture}
```

Now that the fixed points are defined, we can get new points from their references, with the macros of the package or with macros you create yourself. The calculations are usually done by the package and remain hidden. You may need some mathematical constants; here is the list of the constants defined by the package.

7. Auxiliary tools

7.1. Constants

tkz-euclide knows the following constants:

```
\def\tkzPhi{1.6180340}
\def\tkzInvPhi{0.6180340}
\def\tkzSqrtPhi{1.2720020}
\def\tkzSqrTwo{1.414213}
\def\tkzSqrThree{1.7320508}
\def\tkzSqrFive{2.2360679}
\def\tkzSqrTwobyTwo{0.7071065}
\def\tkzPi{3.1415926}
\def\tkzEuler{2.71828182}
```

7.2. New point by calculation

When a macro of **tkz-euclide** creates a new point, it is stored internally with the reference **tkzPointResult**. You can give it your own reference with the macro \tkzGetPoint; the new reference is placed between braces.

\tkzGetPoint{⟨ref⟩}

If the result is in **tkzPointResult**, you can access it with \tkzGetPoint.

arguments	default	example
ref	no default	\tkzGetPoint{M} see the next example

Sometimes you need to get two points. This is done with

\tkzGetPoints{⟨ref1⟩}{⟨ref2⟩}

The results are in **tkzFirstPointResult** and **tkzSecondPointResult**.

arguments	default	example
{ref1,ref2}	no default	\tkzGetPoints{M,N} It's the case with \tkzInterCC

If you only need the first or the second point, you can also use:

\tkzGetFirstPoint{⟨ref1⟩}

arguments	default	example
ref1	no default	\tkzGetFirstPoint{M}

\tkzGetSecondPoint{⟨ref2⟩}

arguments	default	example
ref2	no default	\tkzGetSecondPoint{M}

Sometimes the results consist of a point and a dimension. You get the point with \tkzGetPoint and the dimension with \tkzGetLength.

\tkzGetLength{⟨name of a macro⟩}

arguments	default	example
name of a macro	no default	\tkzGetLength{rAB} \rAB gives the length in cm

8. Special points

Here are some special points.

8.1. Midpoint of a segment \tkzDefMidPoint

This macro defines the midpoint of a segment.

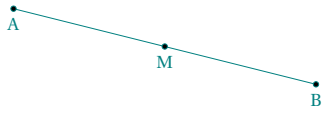
\tkzDefMidPoint(⟨pt1,pt2⟩)

The result is in **tkzPointResult**. We can access it with \tkzGetPoint.

arguments	default	definition
(pt1,pt2)	no default	pt1 and pt2 are two points

8.1.1. Use of \tkzDefMidPoint

See also the use of \tkzDefPoint.




```
\begin{tikzpicture}[scale=1]
 \tkzDefPoint(2,3){A}
 \tkzDefPoint(6,2){B}
 \tkzDefMidPoint(A,B)
 \tkzGetPoint{M}
 \tkzDrawSegment(A,B)
 \tkzDrawPoints(A,B,M)
 \tkzLabelPoints[below](A,B,M)
\end{tikzpicture}
```

8.2. Golden ratio \tkzDefGoldenRatio

From Wikipedia: In mathematics, two quantities are in the golden ratio if their ratio is the same as the ratio of their sum to the larger of the two quantities. Expressed algebraically, for quantities a and b such that a > b > 0, a + b is to a as a is to b.
$\frac{a+b}{a} = \frac{a}{b} = \phi = \frac{1+\sqrt{5}}{2}$
One of the two solutions of the equation $x^2 - x - 1 = 0$ is the golden ratio ϕ, $\phi = \frac{1+\sqrt{5}}{2}$.

\tkzDefGoldenRatio(⟨pt1,pt2⟩)

arguments	default	example
(pt1,pt2)	no default	\tkzDefGoldenRatio(A,C) \tkzGetPoint{B}

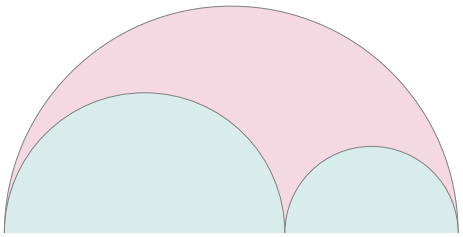
AB = a, BC = b and $\frac{AC}{AB} = \frac{AB}{BC} = \phi$

8.2.1. Use the golden ratio to divide a line segment




```
\begin{tikzpicture}
 \tkzDefPoints{0/0/A,6/0/C}
 \tkzDefMidPoint(A,C) \tkzGetPoint{I}
 %\tkzDefPointWith[linear,K=\tkzInvPhi](A,C)
 \tkzDefGoldenRatio(A,C) \tkzGetPoint{B}
 \tkzDrawSegments(A,C)
 \tkzDrawPoints(A,B,C)
 \tkzLabelPoints(A,B,C)
\end{tikzpicture}
```

8.2.2. Golden arbelos



```
\begin{tikzpicture}[scale=.6]
\tkzDefPoints{0/0/A,10/0/B}
\tkzDefGoldenRatio(A,B)      \tkzGetPoint{C}
\tkzDefMidPoint(A,B)         \tkzGetPoint{O_1}
\tkzDefMidPoint(A,C)         \tkzGetPoint{O_2}
\tkzDefMidPoint(C,B)         \tkzGetPoint{O_3}
\tkzDrawSemiCircles[fill=purple!15](O_1,B)
\tkzDrawSemiCircles[fill=teal!15](O_2,C O_3,B)
\end{tikzpicture}
```

You can also use the following macro.

8.3. Barycentric coordinates with \tkzDefBarycentricPoint

n points pt_1, pt_2, ..., pt_n define n vectors $\vec{v_1}$, $\vec{v_2}$, ..., $\vec{v_n}$ with the origin as common starting point. α_1, α_2, ..., α_n are n numbers; the vector

$$\frac{\alpha_1\vec{v_1}+\alpha_2\vec{v_2}+\cdots+\alpha_n\vec{v_n}}{\alpha_1+\alpha_2+\cdots+\alpha_n}$$

defines a unique point.

\tkzDefBarycentricPoint(⟨pt1=α_1,pt2=α_2,...⟩)

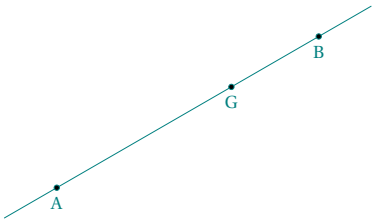
arguments	default	definition
(pt1=α_1,pt2=α_2,...)	no default	Each point has a assigned weight

You need at least two points. Result in **tkzPointResult**.

8.3.1. With two points

In the following example, we get the barycenter of the points A and B with the coefficients 1 and 2, in other words:

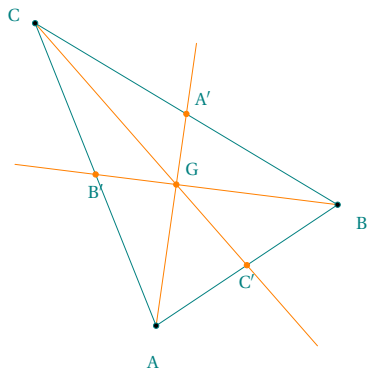
$$\vec{AI}=\frac{2}{3}\vec{AB}$$




```
\begin{tikzpicture}
  \tkzDefPoint(2,3){A}
  \tkzDefShiftPointCoord[2,3](30:4){B}
  \tkzDefBarycentricPoint(A=1,B=2)
  \tkzGetPoint{G}
  \tkzDrawLine(A,B)
  \tkzDrawPoints(A,B,G)
  \tkzLabelPoints(A,B,G)
\end{tikzpicture}
```

8.3.2. With three points

This time M is simply the centroid of the triangle.
For simplicity and consistency, there is also **\tkzCentroid**.




```
\begin{tikzpicture}[scale=.8]
  \tkzDefPoints{2/1/A,5/3/B,0/6/C}
  \tkzDefBarycentricPoint(A=1,B=1,C=1)
  \tkzGetPoint{G}
  \tkzDefMidPoint(A,B)  \tkzGetPoint{C'}
  \tkzDefMidPoint(A,C)  \tkzGetPoint{B'}
  \tkzDefMidPoint(C,B)  \tkzGetPoint{A'}
  \tkzDrawPolygon(A,B,C)
  \tkzDrawLines[add=0 and 1,new](A,G B,G C,G)
  \tkzDrawPoints[new](A',B',C',G)
  \tkzDrawPoints(A,B,C)
  \tkzLabelPoint[above right](G){$G$}
  \tkzAutoLabelPoints[center=G](A,B,C)
  \tkzLabelPoints[above right](A')
  \tkzLabelPoints[below](B',C')
\end{tikzpicture}
```

8.4. Internal and external similitude centers: \tkzDefSimilitudeCenter

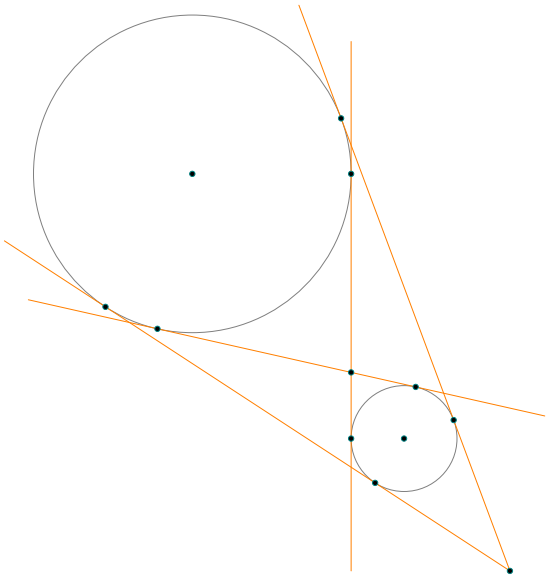
The centers of the two homotheties that map one circle onto another are called the external and internal centers of similitude. You can use **\tkzDefIntSimilitudeCenter** and **\tkzDefExtSimilitudeCenter**, but the following macro is better.

\tkzDefSimilitudeCenter[⟨options⟩](⟨O,A⟩)(⟨O',B⟩)

arguments	example	explanation
(⟨pt1,pt2⟩)(⟨pt3,pt4⟩)	(O,A)(O′,B)	r = OA, r′ = O′B

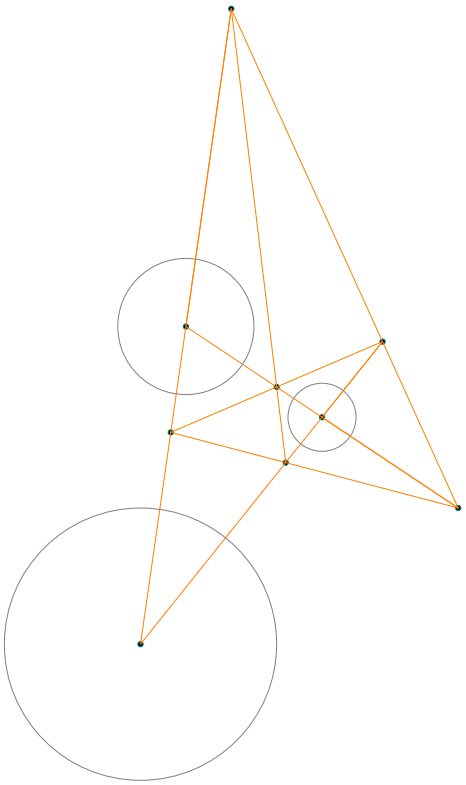
options	default	definition
ext	ext	external center
int	ext	internal center

8.4.1. Internal and external with node



```
\begin{tikzpicture}[scale=.7]
 \tkzDefPoints{0/0/O,4/-5/A,3/0/B,5/-5/C}
 \tkzDefSimilitudeCenter[int](O,B)(A,C)
 \tkzGetPoint{I}
 \tkzDefSimilitudeCenter[ext](O,B)(A,C)
 \tkzGetPoint{J}
 \tkzDefLine[tangent from = I](O,B)
 \tkzGetPoints{D}{E}
 \tkzDefLine[tangent from = I](A,C)
 \tkzGetPoints{D'}{E'}
 \tkzDefLine[tangent from = J](O,B)
 \tkzGetPoints{F}{G}
 \tkzDefLine[tangent from = J](A,C)
 \tkzGetPoints{F'}{G'}
 \tkzDrawCircles(O,B A,C)
 \tkzDrawSegments[add = .5 and .5,new](D,D' E,E')
 \tkzDrawSegments[add= 0 and 0.25,new](J,F J,G)
 \tkzDrawPoints(O,A,I,J,D,E,F,G,D',E',F',G')
\end{tikzpicture}
```

8.4.2. D'Alembert Theorem

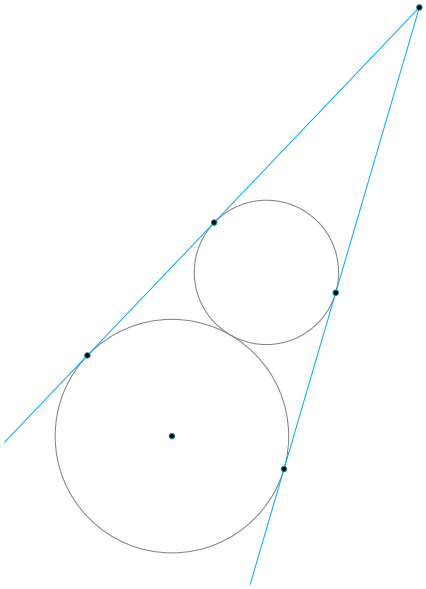


```
\begin{tikzpicture}[scale=.6,rotate=90]
\tkzDefPoints{0/0/A,3/0/a,7/-1/B,5.5/-1/b}
\tkzDefPoints{5/-4/C,4.25/-4/c}
\tkzDrawCircles(A,a B,b C,c)
\tkzDefExtSimilitudeCenter(A,a)(B,b) \tkzGetPoint{I}
\tkzDefExtSimilitudeCenter(A,a)(C,c) \tkzGetPoint{J}
\tkzDefExtSimilitudeCenter(C,c)(B,b) \tkzGetPoint{K}
\tkzDefIntSimilitudeCenter(A,a)(B,b) \tkzGetPoint{I'}
\tkzDefIntSimilitudeCenter(A,a)(C,c) \tkzGetPoint{J'}
\tkzDefIntSimilitudeCenter(C,c)(B,b) \tkzGetPoint{K'}
\tkzDrawPoints(A,B,C,I,J,K,I',J',K')
\tkzDrawSegments[new](I,K A,I A,J B,I B,K C,J C,K)
\tkzDrawSegments[new](I,J' I',J I',K)
\end{tikzpicture}
```

You can also use **\tkzDefBarycentricPoint** to find a center of homothety:

```
\tkzDefBarycentricPoint(O=\r,A=\R)      \tkzGetPoint{I}
\tkzDefBarycentricPoint(O={-\r},A=\R)   \tkzGetPoint{J}
```

8.4.3. Example with node



```
\begin{tikzpicture}[rotate=60,scale=.5]
 \tkzDefPoints{0/0/A,5/0/C}
 \tkzDefGoldenRatio(A,C) \tkzGetPoint{B}
 \tkzDefSimilitudeCenter(A,B)(C,B)\tkzGetPoint{J}
 \tkzDefTangent[from = J](A,B)  \tkzGetPoints{F}{G}
 \tkzDefTangent[from = J](C,B)  \tkzGetPoints{F'}{G'}
 \tkzDrawCircles(A,B C,B)
 \tkzDrawSegments[add= 0 and 0.25,cyan](J,F J,G)
 \tkzDrawPoints(A,J,F,G,F',G')
\end{tikzpicture}
```

8.5. Harmonic division with \tkzDefHarmonic

\tkzDefHarmonic[⟨options⟩](⟨pt1,pt2,pt3⟩) or (⟨pt1,pt2,k⟩)

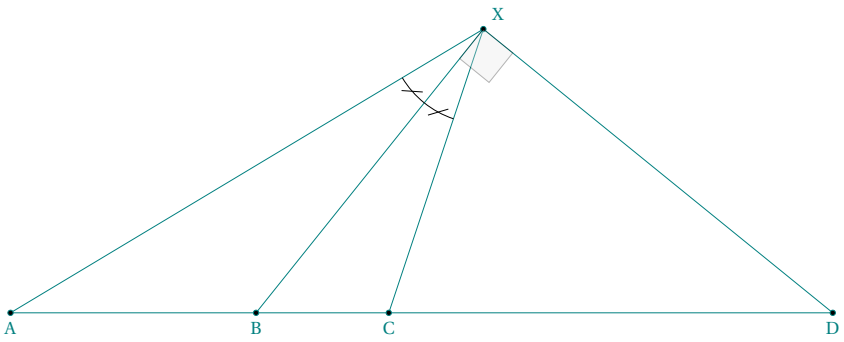
options	default	definition
both	both	(⟨A,B,2⟩) we look for C and D such that (A,B;C,D) = −1 and CA=2CB
ext	both	(⟨A,B,C⟩) we look for D such that (A,B;C,D) = −1
int	both	(⟨A,B,D⟩) we look for C such that (A,B;C,D) = −1

8.5.1. Options ext and int




```
\begin{tikzpicture}
\tkzDefPoints{0/0/A,6/0/B,4/0/C}
\tkzDefHarmonic[ext](A,B,C) \tkzGetPoint{J}
\tkzDefHarmonic[int](A,B,J) \tkzGetPoint{I}
\tkzDrawPoints(A,B,I,J)
\tkzDrawLine[add=.5 and 1](A,B)
\tkzLabelPoints(A,B,I,J)
\end{tikzpicture}
```

8.5.2. Bisector and harmonic division




```
\begin{tikzpicture}[scale=1.25]
\tkzDefPoints{0/0/A,4/0/C,5/3/X}
\tkzDefLine[bisector](A,X,C) \tkzGetPoint{x}
\tkzInterLL(X,x)(A,C)        \tkzGetPoint{B}
\tkzDefHarmonic[ext](A,C,B)  \tkzGetPoint{D}
\tkzDrawPolygon(A,X,C)
\tkzDrawSegments(X,B C,D D,X)
\tkzDrawPoints(A,B,C,D,X)
\tkzMarkAngles[mark=s|](A,X,B B,X,C)
\tkzMarkRightAngle[size=.4,
                   fill=gray!20,
                   opacity=.3](B,X,D)
\tkzLabelPoints(A,B,C,D)
\tkzLabelPoints[above right](X)
\end{tikzpicture}
```

8.5.3. Option both

both is the default option




```
\begin{tikzpicture}
 \tkzDefPoints{0/0/A,6/0/B}
 \tkzDefHarmonic(A,B,{1/2})\tkzGetPoints{I}{J}
 \tkzDrawPoints(A,B,I,J)
 \tkzDrawLine[add=1 and .5](A,B)
 \tkzLabelPoints(A,B,I,J)
\end{tikzpicture}
```

8.6. Equidistant points with \tkzDefEquiPoints

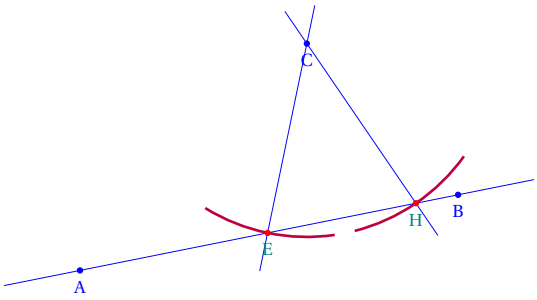
\tkzDefEquiPoints[⟨local options⟩](⟨pt1,pt2⟩)

arguments	default	definition
(pt1,pt2)	no default	unordered list of two items

options	default	definition
dist	2 (cm)	half the distance between the two points
from=pt	no default	reference point
show	false	if true displays compass traces
/compass/delta	0	size of the compass mark

This macro gives two points of a straight line equidistant from a given point.

8.6.1. Using \tkzDefEquiPoints with options

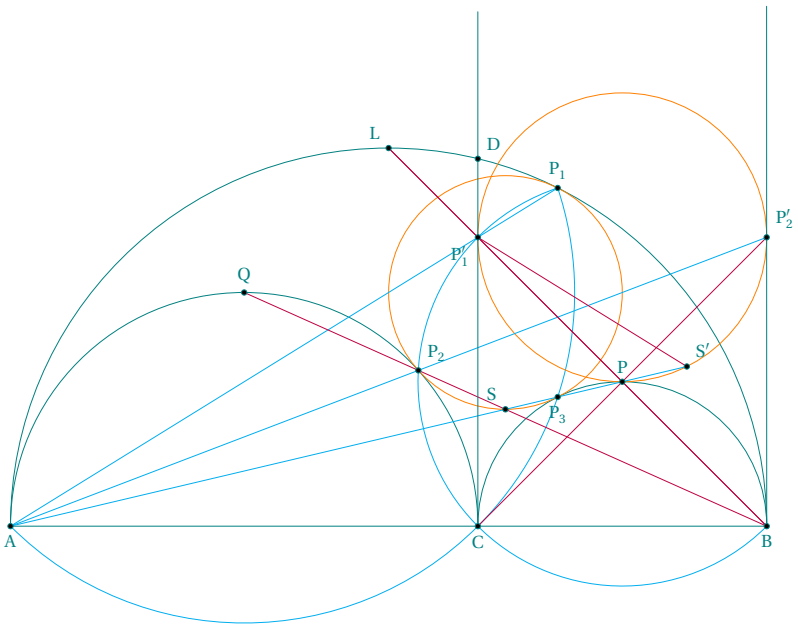



```
\begin{tikzpicture}
  \tkzSetUpCompass[color=purple,line width=1pt]
  \tkzDefPoints{0/1/A,5/2/B,3/4/C}
  \tkzDefEquiPoints[from=C,dist=1,show,
      /tkzcompass/delta=20](A,B)
   \tkzGetPoints{E}{H}
   \tkzDrawLines[color=blue](C,E C,H A,B)
   \tkzDrawPoints[color=blue](A,B,C)
   \tkzDrawPoints[color=red](E,H)
   \tkzLabelPoints(E,H)
   \tkzLabelPoints[color=blue](A,B,C)
\end{tikzpicture}
```

8.7. Midpoint of an arc

\tkzDefMidArc(⟨pt1,pt2,pt3⟩)

arguments	default	definition
pt1,pt2,pt3	no default	pt1 is the center, $\overset{\frown}{\text{pt2pt3}}$ the arc


L
D
P1
P'2
P'1
Q
P2
S'
P
S
P3
A
C
B

```
\begin{tikzpicture}[scale=1]
 \tkzDefPoints{0/0/A,10/0/B}
 \tkzDefGoldenRatio(A,B)                                  \tkzGetPoint{C}
 \tkzDefMidPoint(A,B)                                     \tkzGetPoint{O_1}
 \tkzDefMidPoint(A,C)                                     \tkzGetPoint{O_2}
 \tkzDefMidPoint(C,B)                                     \tkzGetPoint{O_3}
 \tkzDefMidArc(O_3,B,C)                                   \tkzGetPoint{P}
 \tkzDefMidArc(O_2,C,A)                                   \tkzGetPoint{Q}
 \tkzDefMidArc(O_1,B,A)                                   \tkzGetPoint{L}
 \tkzDefPointBy[rotation=center C angle 90](B)            \tkzGetPoint{c}
 \tkzInterCC[common=B](P,B)(O_1,B)                        \tkzGetFirstPoint{P_1}
 \tkzInterCC[common=C](P,C)(O_2,C)                        \tkzGetFirstPoint{P_2}
 \tkzInterCC[common=C](Q,C)(O_3,C)                        \tkzGetFirstPoint{P_3}
 \tkzInterLC[near](c,C)(O_1,A)                            \tkzGetFirstPoint{D}
 \tkzInterLL(A,P_1)(C,D)                                  \tkzGetPoint{P_1'}
 \tkzDefPointBy[inversion = center A through D](P_2)      \tkzGetPoint{P_2'}
 \tkzDefCircle[circum](P_3,P_2,P_1)                       \tkzGetPoint{O_4}
 \tkzInterLL(B,Q)(A,P)                                    \tkzGetPoint{S}
 \tkzDefMidPoint(P_2',P_1')                               \tkzGetPoint{o}
 \tkzDefPointBy[inversion = center A through D](S)        \tkzGetPoint{S'}
 \tkzDrawArc[cyan,delta=0](Q,A)(P_1)
 \tkzDrawArc[cyan,delta=0](P,P_1)(B)
 \tkzDrawSemiCircles[teal](O_1,B O_2,C O_3,B)
 \tkzDrawCircles[new](o,P O_4,P_1)
 \tkzDrawSegments(A,B)
 \tkzDrawSegments[cyan](A,P_1 A,S' A,P_2')
 \tkzDrawSegments[purple](B,L C,P_2' B,Q B,L S',P_1')
 \tkzDrawLines[add=0 and .8](B,P_2')
 \tkzDrawLines[add=0 and .4](C,D)
 \tkzDrawPoints(A,B,C,P,Q,P_3,P_2,P_1,P_1',D,P_2',L,S,S')
 \tkzLabelPoints(A,B,C,P_3)
 \tkzLabelPoints[above](P,Q,P_1)
 \tkzLabelPoints[above right](P_2,P_2',D,S')
 \tkzLabelPoints[above left](L,S)
  \tkzLabelPoints[below left](P_1')
\end{tikzpicture}
```

9. Point on line or circle

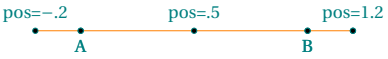
9.1. Point on a line with \tkzDefPointOnLine

\tkzDefPointOnLine[⟨local options⟩](⟨A,B⟩)

arguments	default	definition
pt1,pt2	no default	Two points to define a line

options	default	definition
pos=nb		nb is a decimal

9.1.1. Use of option pos




```
\begin{tikzpicture}
\tkzDefPoints{0/0/A,3/0/B}
\tkzDefPointOnLine[pos=1.2](A,B)\tkzGetPoint{P}
\tkzDefPointOnLine[pos=-0.2](A,B)\tkzGetPoint{R}
\tkzDefPointOnLine[pos=0.5](A,B) \tkzGetPoint{S}
\tkzDrawLine[new](A,B)
\tkzDrawPoints(A,B,P)
\tkzLabelPoints(A,B)
\tkzLabelPoint[above](P){pos=$1.2$}
\tkzLabelPoint[above](R){pos=$-.2$}
\tkzLabelPoint[above](S){pos=$.5$}
\tkzDrawPoints(A,B,P,R,S)
\tkzLabelPoints(A,B)
\end{tikzpicture}
```

9.2. Point on a circle with \tkzDefPointOnCircle

The order of the arguments has changed: now it is center, angle and point or radius. Two options work with radians: **through in rad** and **R in rad**.

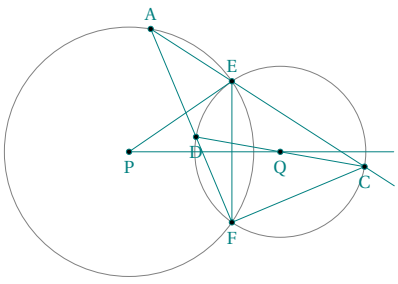
\tkzDefPointOnCircle[⟨local options⟩]

options	default	examples definition
through		through = center K1 angle 30 point B]
R		R = center K1 angle 30 radius \rAp
through in rad		through in rad= center K1 angle pi/4 point B]
R in rad		R in rad = center K1 angle pi/6 radius \rAp

The order of the arguments is: center, angle, and point or radius.

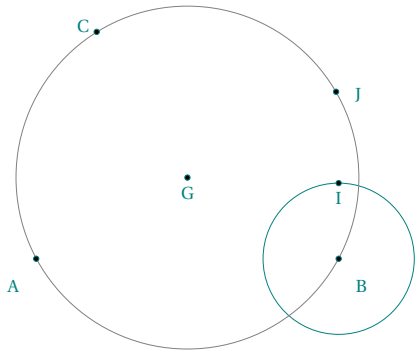
9.2.1. Altshiller's Theorem

The two lines joining the points of intersection of two orthogonal circles to a point of one of the circles meet the other circle in two diametrically opposite points (Altshiller-Court, p. 176).




```
\begin{tikzpicture}[scale=.4]
\tkzDefPoints{0/0/P,5/0/Q,3/2/I}
\tkzDefCircle[orthogonal from=P](Q,I)
\tkzGetFirstPoint{E}
\tkzDrawCircles(P,E Q,E)
\tkzInterCC[common=E](P,E)(Q,E) \tkzGetFirstPoint{F}
\tkzDefPointOnCircle[through =  center P angle 80 point E]
 \tkzGetPoint{A}
\tkzInterLC[common=E](A,E)(Q,E)  \tkzGetFirstPoint{C}
\tkzInterLL(A,F)(C,Q)  \tkzGetPoint{D}
\tkzDrawLines[add=0 and .75](P,Q)
\tkzDrawLines[add=0 and 2](A,E)
\tkzDrawSegments(P,E E,F F,C A,F C,D)
\tkzDrawPoints(P,Q,E,F,A,C,D)
\tkzLabelPoints(P,Q,F,C,D)
\tkzLabelPoints[above](E,A)
\end{tikzpicture}
```

9.2.2. Use of \tkzDefPointOnCircle




```
\begin{tikzpicture}
\tkzDefPoints{0/0/A,4/0/B,0.8/3/C}
\tkzDefPointOnCircle[R = center B  angle 90 radius 1]
\tkzGetPoint{I}
\tkzDefCircle[circum](A,B,C)
\tkzGetPoints{G}{g}
\tkzDefPointOnCircle[through = center G angle 30 point g]
\tkzGetPoint{J}
\tkzDefCircle[R](B,1) \tkzGetPoint{b}
\tkzDrawCircle[teal](B,b)
\tkzDrawCircle(G,J)
\tkzDrawPoints(A,B,C,G,I,J)
\tkzAutoLabelPoints[center=G](A,B,C,J)
\tkzLabelPoints[below](G,I)
\end{tikzpicture}
```

10. Special points relating to a triangle

10.1. Triangle center: \tkzDefTriangleCenter

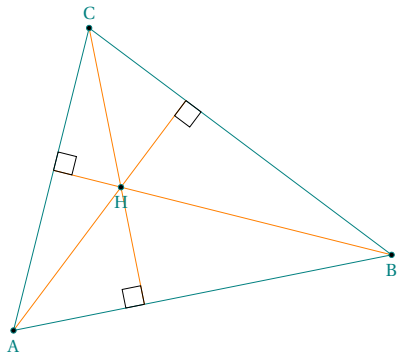
\tkzDefTriangleCenter[⟨local options⟩](⟨A,B,C⟩)

This macro defines a center of a triangle. Be careful: the arguments are lists of three points. The macro is used together with \tkzGetPoint to get the center you are looking for.
You can use tkzPointResult directly if you don't need to keep the result.

arguments	default	example
(pt1,pt2,pt3)	no default	\tkzDefTriangleCenter[ortho](B,C,A)
options	default	definition
ortho	circum	intersection of the altitudes
orthic	circum	…
centroid	circum	intersection of the medians
median	circum	…
circum	circum	circle center circumscribed
in	circum	center of the circle inscribed in a triangle
in	circum	intersection of the bisectors
ex	circum	center of a circle exinscribed to a triangle
euler	circum	center of Euler's circle
gergonne	circum	defined with the Contact triangle
symmedian	circum	Lemoine's point or symmedian center or Grebe's point
lemoine	circum	…
grebe	circum	…
spieker	circum	Spieker circle center
nagel	circum	Nagel Center
mittenpunkt	circum	Or middlespoint
feuerbach	circum	Feuerbach Point

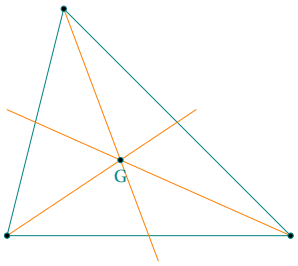
10.1.1. Option ortho or orthic

The intersection H of the three altitudes of a triangle is called the orthocenter.



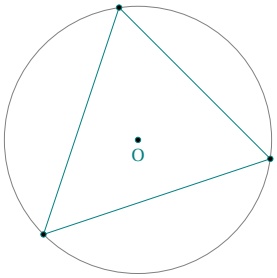

```
\begin{tikzpicture}
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(5,1){B}
  \tkzDefPoint(1,4){C}
  \tkzDefTriangleCenter[ortho](B,C,A)
  \tkzGetPoint{H}
  \tkzDefSpcTriangle[orthic,name=H](A,B,C){a,b,c}
  \tkzDrawPolygon(A,B,C)
  \tkzDrawSegments[new](A,Ha B,Hb C,Hc)
  \tkzDrawPoints(A,B,C,H)
  \tkzLabelPoint(H){$H$}
  \tkzLabelPoints[below](A,B)
  \tkzLabelPoints[above](C)
 \tkzMarkRightAngles(A,Ha,B B,Hb,C C,Hc,A)
\end{tikzpicture}
```

10.1.2. Option centroid




```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoints{0/0/A,5/0/B,1/4/C}
  \tkzDefTriangleCenter[centroid](A,B,C)
  \tkzGetPoint{G}
  \tkzDrawPolygon(A,B,C)
  \tkzDrawLines[add = 0 and 2/3,new](A,G B,G C,G)
  \tkzDrawPoints(A,B,C,G)
  \tkzLabelPoint(G){$G$}
\end{tikzpicture}
```

10.1.3. Option circum

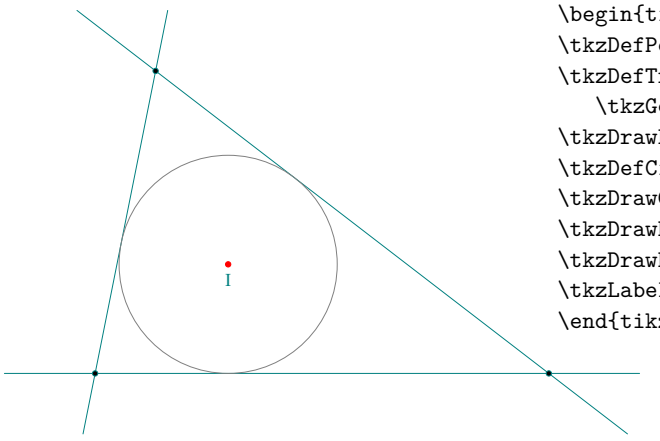



```
 \begin{tikzpicture}
  \tkzDefPoints{0/1/A,3/2/B,1/4/C}
  \tkzDefTriangleCenter[circum](A,B,C)
  \tkzGetPoint{O}
  \tkzDrawPolygon(A,B,C)
  \tkzDrawCircle(O,A)
  \tkzDrawPoints(A,B,C,O)
  \tkzLabelPoint(O){$O$}
\end{tikzpicture}
```

10.1.4. Option in

In geometry, the incircle or inscribed circle of a triangle is the largest circle contained in the triangle; it touches (is tangent to) the three sides. The center of the incircle is a triangle center called the triangle's incenter. The center of the incircle, called the incenter, can be found as the intersection of the three internal angle bisectors. The center of an excircle is the intersection of the internal bisector of one angle (at vertex A, for example) and the external bisectors of the other two. The center of this excircle is called the excenter relative to the vertex A, or the excenter of A. Because the internal bisector of an angle is perpendicular to its external bisector, it follows that the center of the incircle together with the three excircle centers form an orthocentric system.
(Article on Wikipedia)

We get the center of the incircle of the triangle. The result is of course in **tkzPointResult**; we retrieve it with **\tkzGetPoint**.

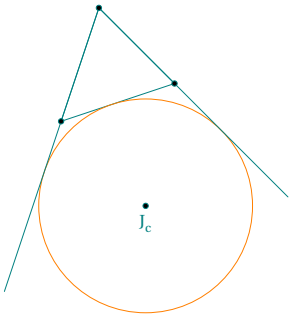



```
\begin{tikzpicture}
\tkzDefPoints{0/0/A,6/0/B,0.8/4/C}
\tkzDefTriangleCenter[in](A,B,C)
   \tkzGetPoint{I}
\tkzDrawLines(A,B B,C C,A)
\tkzDefCircle[in](A,B,C) \tkzGetPoints{I}{i}
\tkzDrawCircle(I,i)
\tkzDrawPoint[red](I)
\tkzDrawPoints(A,B,C)
\tkzLabelPoint(I){$I$}
\end{tikzpicture}
```

10.1.5. Option ex

An excircle or escribed circle of the triangle is a circle lying outside the triangle, tangent to one of its sides and tangent to the extensions of the other two. Every triangle has three distinct excircles, each tangent to one of the

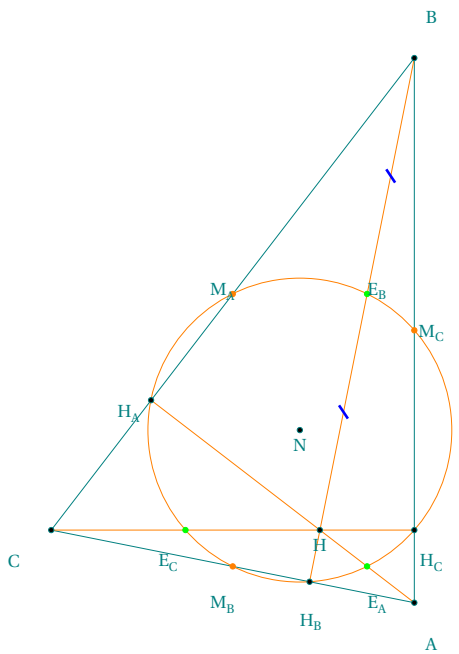
triangle's sides.
(Article on Wikipedia)
We get the center of an excircle of the triangle. The result is of course in **tkzPointResult**; we retrieve it with **\tkzGetPoint**.




```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoints{0/1/A,3/2/B,1/4/C}
  \tkzDefTriangleCenter[ex](B,C,A)
  \tkzGetPoint{J_c}
  \tkzDefPointBy[projection=onto A--B](J_c)
  \tkzGetPoint{Tc}
  \tkzDrawPolygon(A,B,C)
  \tkzDrawCircle[new](J_c,Tc)
  \tkzDrawLines[add=1.5 and 0](A,C B,C)
  \tkzDrawPoints(A,B,C,J_c)
  \tkzLabelPoints(J_c)
\end{tikzpicture}
```

10.1.6. Option euler

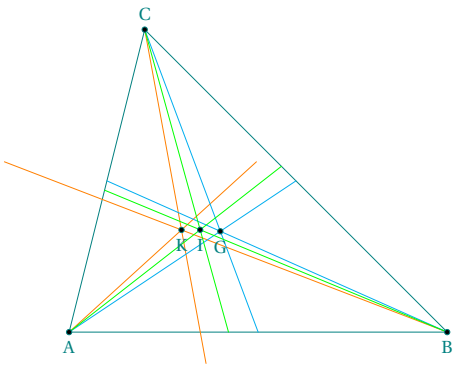
This macro gives the center of the nine-point circle, also called Euler's circle or Feuerbach's circle. The nine-point circle, also called Euler's circle or the Feuerbach circle, is the circle that passes through the perpendicular feet H_A, H_B, and H_C dropped from the vertices of any reference triangle ABC on the sides opposite them. Euler showed in 1765 that it also passes through the midpoints M_A, M_B, M_C of the sides of ABC. By Feuerbach's theorem, the nine-point circle also passes through the midpoints E_A, E_B, and E_C of the segments that join the vertices and the orthocenter H. These points are commonly referred to as the Euler points.
(https://mathworld.wolfram.com/Nine-PointCircle.html)




```
\begin{tikzpicture}[scale=1.2,rotate=90]
 \tkzDefPoints{0/0/A,6/0/B,0.8/4/C}
 \tkzDefSpcTriangle[medial,name=M](A,B,C){_A,_B,_C}
 \tkzDefTriangleCenter[euler](A,B,C)\tkzGetPoint{N}
 % I= N nine points
 \tkzDefTriangleCenter[ortho](A,B,C)\tkzGetPoint{H}
 \tkzDefMidPoint(A,H) \tkzGetPoint{E_A}
 \tkzDefMidPoint(C,H) \tkzGetPoint{E_C}
 \tkzDefMidPoint(B,H) \tkzGetPoint{E_B}
 \tkzDefSpcTriangle[ortho,name=H](A,B,C){_A,_B,_C}
 \tkzDrawPolygon(A,B,C)
 \tkzDrawCircle[new](N,E_A)
 \tkzDrawSegments[new](A,H_A B,H_B C,H_C)
 \tkzDrawPoints(A,B,C,N,H)
 \tkzDrawPoints[new](M_A,M_B,M_C)
 \tkzDrawPoints( H_A,H_B,H_C)
 \tkzDrawPoints[green](E_A,E_B,E_C)
 \tkzAutoLabelPoints[center=N,
 font=\scriptsize](A,B,C,M_A,M_B,M_C,H_A,H_B,H_C,%
   E_A,E_B,E_C)
 \tkzLabelPoints[font=\scriptsize](H,N)
 \tkzMarkSegments[mark=s|,size=3pt,
 color=blue,line width=1pt](B,E_B E_B,H)
\end{tikzpicture}
```

10.1.7. Option symmedian

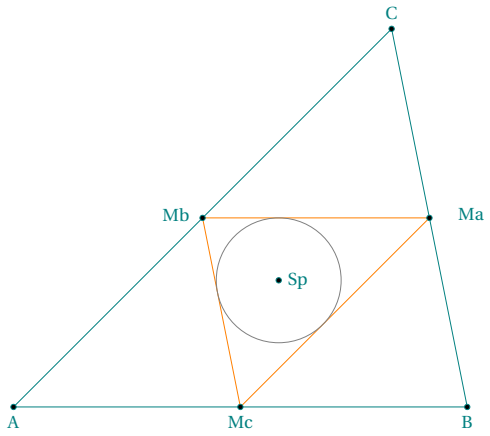
The point of concurrence K of the symmedians, sometimes also called the Lemoine point (in England and France) or the Grebe point (in Germany).
Weisstein, Eric W. "Symmedian Point." From MathWorld–A Wolfram Web Resource.




```
\begin{tikzpicture}
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(5,0){B}
  \tkzDefPoint(1,4){C}
  \tkzDefTriangleCenter[symmedian](A,B,C)
  \tkzGetPoint{K}
  \tkzDefTriangleCenter[median](A,B,C)
  \tkzGetPoint{G}
  \tkzDefTriangleCenter[in](A,B,C)\tkzGetPoint{I}
  \tkzDefSpcTriangle[centroid,name=M](A,B,C){a,b,c}
  \tkzDefSpcTriangle[incentral,name=I](A,B,C){a,b,c}
  \tkzDrawPolygon(A,B,C)
  \tkzDrawLines[add = 0 and 2/3,new](A,K B,K C,K)
  \tkzDrawSegments[color=cyan](A,Ma B,Mb C,Mc)
  \tkzDrawSegments[color=green](A,Ia B,Ib C,Ic)
  \tkzDrawPoints(A,B,C,K,G,I)
  \tkzLabelPoints[font=\scriptsize](A,B,K,G,I)
  \tkzLabelPoints[above,font=\scriptsize](C)
\end{tikzpicture}
```

10.1.8. Option spieker

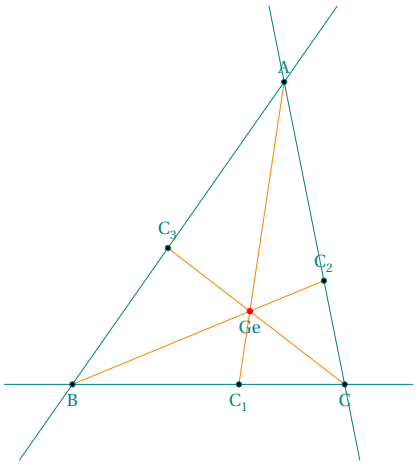
The Spieker center is the center Sp of the Spieker circle, i.e., the incenter of the medial triangle of a reference triangle.
Weisstein, Eric W. "Spieker Center." From MathWorld–A Wolfram Web Resource.




```
\begin{tikzpicture}
 \tkzDefPoints{0/0/A,6/0/B,5/5/C}
 \tkzDefSpcTriangle[medial](A,B,C){Ma,Mb,Mc}
 \tkzDefTriangleCenter[centroid](A,B,C)
 \tkzGetPoint{G}
 \tkzDefTriangleCenter[spieker](A,B,C)
 \tkzGetPoint{Sp}
 \tkzDrawPolygon[](A,B,C)
 \tkzDrawPolygon[new](Ma,Mb,Mc)
 \tkzDefCircle[in](Ma,Mb,Mc) \tkzGetPoints{I}{i}
 \tkzDrawCircle(I,i)
 \tkzDrawPoints(B,C,A,Sp,Ma,Mb,Mc)
 \tkzAutoLabelPoints[center=G,dist=.3](Ma,Mb)
 \tkzLabelPoints[right](Sp)
 \tkzLabelPoints[below](A,B,Mc)
 \tkzLabelPoints[above](C)
\end{tikzpicture}
```

10.1.9. Option gergonne

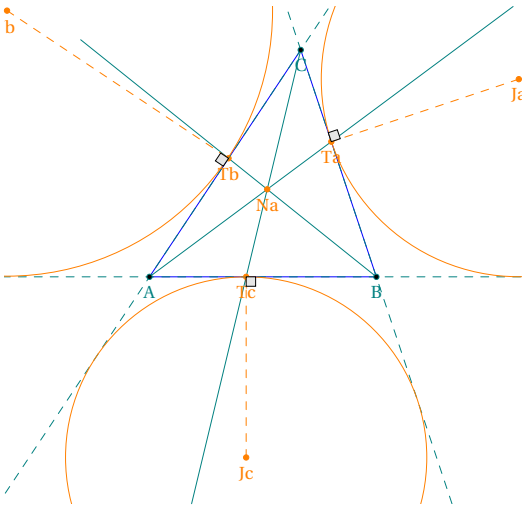
The Gergonne Point is the point of concurrency which results from connecting the vertices of a triangle to the opposite points of tangency of the triangle's incircle. (Joseph Gergonne, French mathematician.)




```
\begin{tikzpicture}
\tkzDefPoints{0/0/B,3.6/0/C,2.8/4/A}
\tkzDefTriangleCenter[gergonne](A,B,C)
\tkzGetPoint{Ge}
\tkzDefSpcTriangle[intouch](A,B,C){C_1,C_2,C_3}
\tkzDefCircle[in](A,B,C) \tkzGetPoints{I}{i}
\tkzDrawLines[add=.25 and .25,teal](A,B A,C B,C)
\tkzDrawSegments[new](A,C_1 B,C_2 C,C_3)
\tkzDrawPoints(A,...,C,C_1,C_2,C_3)
\tkzDrawPoints[red](Ge)
\tkzLabelPoints(B,C,C_1,Ge)
\tkzLabelPoints[above](A,C_2,C_3)
\end{tikzpicture}
```

10.1.10. Option nagel

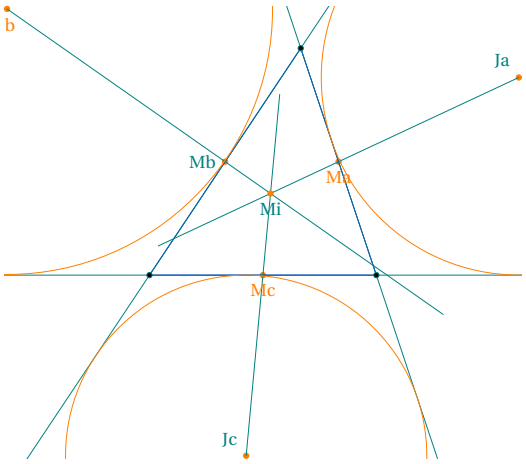
Let Ta be the point at which the excircle with center Ja meets the side BC of a triangle ABC, and define Tb and Tc similarly. Then the lines ATa, BTb, and CTc concur in the Nagel point Na.
Weisstein, Eric W. "Nagel point." From MathWorld–A Wolfram Web Resource.




```
\begin{tikzpicture}[scale=.5]
\tkzDefPoints{0/0/A,6/0/B,4/6/C}
\tkzDefSpcTriangle[ex](A,B,C){Ja,Jb,Jc}
\tkzDefSpcTriangle[extouch](A,B,C){Ta,Tb,Tc}
\tkzDefTriangleCenter[nagel](A,B,C)
\tkzGetPoint{Na}
\tkzDrawPolygon[blue](A,B,C)
\tkzDrawLines[add=0 and 1](A,Ta B,Tb C,Tc)
\tkzDrawPoints[new](Ja,Jb,Jc,Ta,Tb,Tc)
\tkzClipBB
\tkzDrawLines[add=1 and 1,dashed](A,B B,C C,A)
\tkzDrawCircles[new](Ja,Ta Jb,Tb Jc,Tc)
\tkzDrawSegments[new,dashed](Ja,Ta Jb,Tb Jc,Tc)
\tkzDrawPoints(B,C,A)
\tkzDrawPoints[new](Na)
\tkzLabelPoints(B,C,A)
\tkzLabelPoints[new](Na)
\tkzLabelPoints[new](Ja,Jb,Jc,Ta,Tb,Tc)
\tkzMarkRightAngles[fill=gray!20](Ja,Ta,C
            Jb,Tb,A Jc,Tc,B)
\end{tikzpicture}
```

10.1.11. Option mittenpunkt

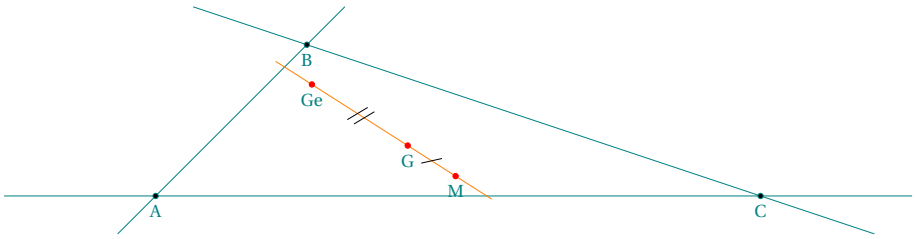
The mittenpunkt (also called the middlespoint) of a triangle ABC is the symmedian point of the excentral triangle, i.e., the point of concurrence M of the lines from the excenters through the midpoints of the corresponding sides.
Weisstein, Eric W. "Mittenpunkt." From MathWorld–A Wolfram Web Resource.




```
\begin{tikzpicture}[scale=.5]
 \tkzDefPoints{0/0/A,6/0/B,4/6/C}
 \tkzDefSpcTriangle[centroid](A,B,C){Ma,Mb,Mc}
 \tkzDefSpcTriangle[ex](A,B,C){Ja,Jb,Jc}
 \tkzDefSpcTriangle[extouch](A,B,C){Ta,Tb,Tc}
 \tkzDefTriangleCenter[mittenpunkt](A,B,C)
 \tkzGetPoint{Mi}
 \tkzDrawPoints[new](Ma,Mb,Mc,Ja,Jb,Jc)
 \tkzClipBB
 \tkzDrawPolygon[blue](A,B,C)
 \tkzDrawLines[add=0 and 1](Ja,Ma
                Jb,Mb Jc,Mc)
 \tkzDrawLines[add=1 and 1](A,B A,C B,C)
 \tkzDrawCircles[new](Ja,Ta Jb,Tb Jc,Tc)
 \tkzDrawPoints(B,C,A)
 \tkzDrawPoints[new](Mi)
 \tkzLabelPoints(Mi)
 \tkzLabelPoints[left](Mb)
 \tkzLabelPoints[new](Ma,Mc,Jb,Jc)
 \tkzLabelPoints[above left](Ja,Jc)
\end{tikzpicture}
```

10.1.12. Relation between gergonne, centroid and mittenpunkt

The Gergonne point Ge, the centroid G and the mittenpunkt M are collinear, with GeG/GM = 2.




```
\begin{tikzpicture}
\tkzDefPoints{0/0/A,2/2/B,8/0/C}
\tkzDefTriangleCenter[gergonne](A,B,C) \tkzGetPoint{Ge}
\tkzDefTriangleCenter[centroid](A,B,C)
\tkzGetPoint{G}
\tkzDefTriangleCenter[mittenpunkt](A,B,C)
\tkzGetPoint{M}
\tkzDrawLines[add=.25 and .25,teal](A,B A,C B,C)
\tkzDrawLines[add=.25 and .25,new](Ge,M)
\tkzDrawPoints(A,...,C)
\tkzDrawPoints[red,size=2](G,M,Ge)
\tkzLabelPoints(A,...,C,M,G,Ge)
\tkzMarkSegment[mark=s||](Ge,G)
\tkzMarkSegment[mark=s|](G,M)
\end{tikzpicture}
```

11. Definition of points by transformation

These transformations are:

- translation;
- homothety;

- orthogonal reflection or symmetry;
- central symmetry;
- orthogonal projection;
- rotation (degrees or radians);
- inversion with respect to a circle.

11.1. \tkzDefPointBy

The transformation is chosen with the options. There are two macros: **\tkzDefPointBy** transforms a single point, and **\tkzDefPointsBy** a list of points. By default, the image of A is A′. For example, we write:

```
\tkzDefPointBy[translation= from A to A'](B)
```

The result is in **tkzPointResult**.

\tkzDefPointBy[⟨local options⟩](⟨pt⟩)

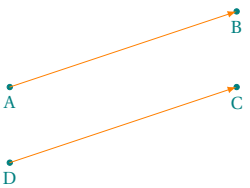
The argument is an existing point, and its image is stored in **tkzPointResult**. To keep this point, **\tkzGetPoint{M}** gives it the name **M**.

arguments	definition	examples
pt	existing point name	(A)

options		examples
translation	= from #1 to #2	[translation=from A to B](E)
homothety	= center #1 ratio #2	[homothety=center A ratio .5](E)
reflection	= over #1--#2	[reflection=over A--B](E)
symmetry	= center #1	[symmetry=center A](E)
projection	= onto #1--#2	[projection=onto A--B](E)
rotation	= center #1 angle #2	[rotation=center O angle 30](E)
rotation in rad	= center #1 angle #2	[rotation in rad=center O angle pi/3](E)
rotation with nodes	= center #1 from #2 to #3	[center O from A to B](E)
inversion	= center #1 through #2	[inversion =center O through A](E)
inversion negative	= center #1 through #2	...

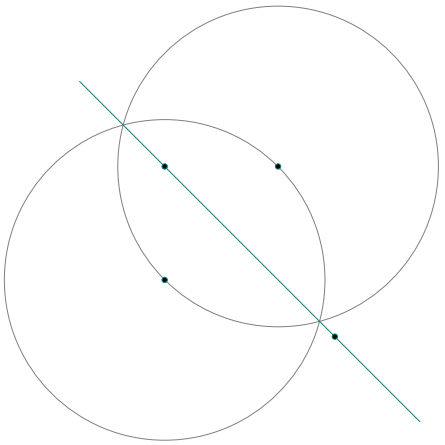
The image is only defined, not drawn.

11.1.1. translation



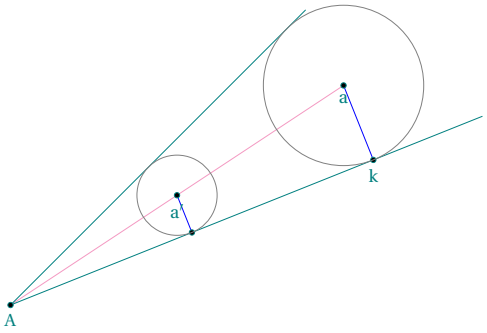

```
\begin{tikzpicture}[>=latex]
 \tkzDefPoints{0/0/A,3/1/B,3/0/C}
 \tkzDefPointBy[translation= from B to A](C)
 \tkzGetPoint{D}
 \tkzDrawPoints[teal](A,B,C,D)
 \tkzLabelPoints[color=teal](A,B,C,D)
 \tkzDrawSegments[orange,->](A,B D,C)
\end{tikzpicture}
```

11.1.2. reflection (orthogonal symmetry)



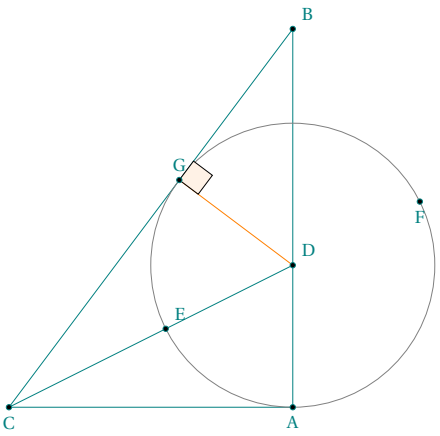
```
\begin{tikzpicture}[scale=.75]
 \tkzDefPoints{-2/-2/A,-1/-1/C,-4/2/D,-4/0/O}
 \tkzDrawCircle(O,A)
 \tkzDefPointBy[reflection = over C--D](A)
 \tkzGetPoint{A'}
 \tkzDefPointBy[reflection = over C--D](O)
 \tkzGetPoint{O'}
 \tkzDrawCircle(O',A')
 \tkzDrawLine[add= .5 and .5](C,D)
 \tkzDrawPoints(C,D,O,O')
\end{tikzpicture}
```

11.1.3. homothety and projection



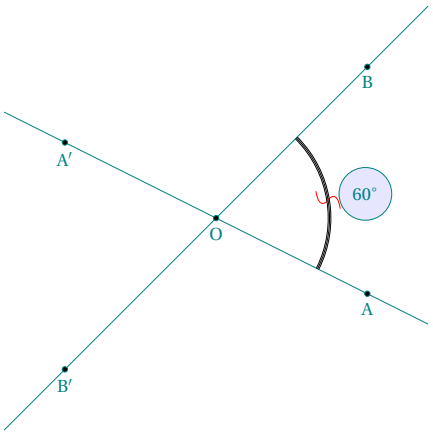

```
\begin{tikzpicture}
  \tkzDefPoints{0/1/A,5/3/B,3/4/C}
  \tkzDefLine[bisector](B,A,C) \tkzGetPoint{a}
  \tkzDrawLine[add=0 and 0,color=magenta!50 ](A,a)
  \tkzDefPointBy[homothety=center A ratio .5](a)
  \tkzGetPoint{a'}
  \tkzDefPointBy[projection = onto A--B](a')
  \tkzGetPoint{k'}
  \tkzDefPointBy[projection = onto A--B](a)
  \tkzGetPoint{k}
  \tkzDrawLines[add= 0 and .3](A,k A,C)
  \tkzDrawSegments[blue](a',k' a,k)
  \tkzDrawPoints(a,a',k,k',A)
  \tkzDrawCircles(a',k' a,k)
  \tkzLabelPoints(a,a',k,A)
\end{tikzpicture}
```

11.1.4. projection



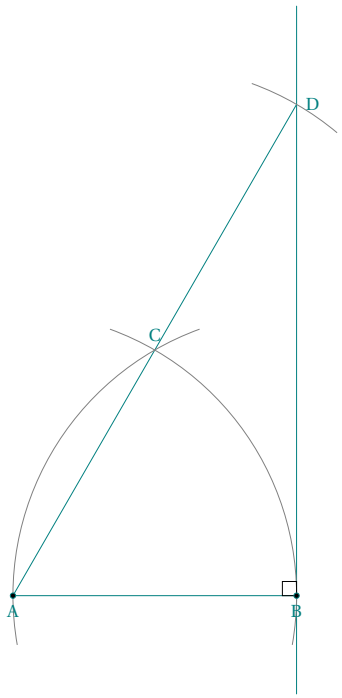

```
\begin{tikzpicture}[scale=1.25]
 \tkzDefPoints{0/0/A,0/4/B}
 \tkzDefTriangle[pythagore](B,A) \tkzGetPoint{C}
 \tkzDefLine[bisector](B,C,A) \tkzGetPoint{c}
 \tkzInterLL(C,c)(A,B)  \tkzGetPoint{D}
 \tkzDefPointBy[projection=onto B--C](D)
 \tkzGetPoint{G}
 \tkzInterLC(C,D)(D,A) \tkzGetPoints{E}{F}
 \tkzDrawPolygon(A,B,C)
 \tkzDrawSegment(C,D)
 \tkzDrawCircle(D,A)
 \tkzDrawSegment[new](D,G)
 \tkzMarkRightAngle[fill=orange!10](D,G,B)
 \tkzDrawPoints(A,C,F) \tkzLabelPoints(A,C,F)
 \tkzDrawPoints(B,D,E,G)
 \tkzLabelPoints[above right](B,D,E)
  \tkzLabelPoints[above](G)
 \end{tikzpicture}
```

11.1.5. symmetry



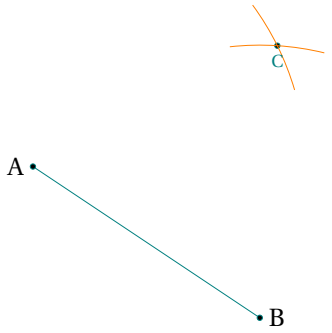

```
\begin{tikzpicture}[scale=1]
  \tkzDefPoints{2/-1/A,2/2/B,0/0/O}
  \tkzDefPointsBy[symmetry=center O](B,A){}
  \tkzDrawLine(A,A')
  \tkzDrawLine(B,B')
  \tkzMarkAngle[mark=s,arc=lll,
      size=1.5,mkcolor=red](A,O,B)
  \tkzLabelAngle[pos=2,circle,draw,
    fill=blue!10,font=\scriptsize](A,O,B){$60^{\circ}$}
  \tkzDrawPoints(A,B,O,A',B')
  \tkzLabelPoints(B,B')
  \tkzLabelPoints[below](A,O,A')
\end{tikzpicture}
```

11.1.6. rotation



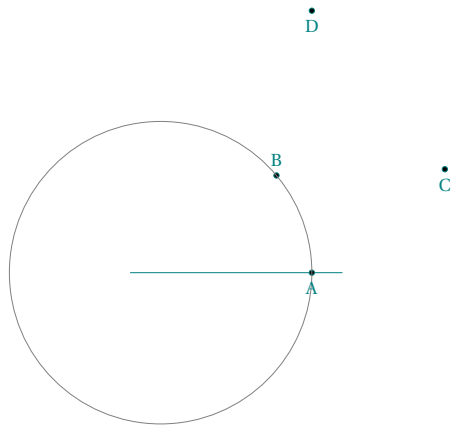

```
\begin{tikzpicture}[scale=0.75]
 \tkzDefPoints{0/0/A,5/0/B}
 \tkzDrawSegment(A,B)
 \tkzDefPointBy[rotation=center A angle 60](B)
 \tkzGetPoint{C}
 \tkzDefPointBy[symmetry=center C](A)
 \tkzGetPoint{D}
 \tkzDrawSegment(A,tkzPointResult)
 \tkzDrawLine(B,D)
 \tkzDrawArc(A,B)(C) \tkzDrawArc(B,C)(A)
 \tkzDrawArc(C,D)(D)
 \tkzMarkRightAngle(D,B,A)
 \tkzDrawPoints(A,B)
 \tkzLabelPoints(A,B)
 \tkzLabelPoints[above](C)
 \tkzLabelPoints[right](D)
\end{tikzpicture}
```

11.1.7. rotation in radian




```
\begin{tikzpicture}
  \tkzDefPoint["$A$" left](1,5){A}
  \tkzDefPoint["$B$" right](4,3){B}
  \tkzDefPointBy[rotation in rad= center A angle pi/3](B)
  \tkzGetPoint{C}
  \tkzDrawSegment(A,B)
  \tkzDrawPoints(A,B,C)
  \tkzCompass(A,C)
  \tkzCompass(B,C)
  \tkzLabelPoints(C)
\end{tikzpicture}
```

11.1.8. rotation with nodes



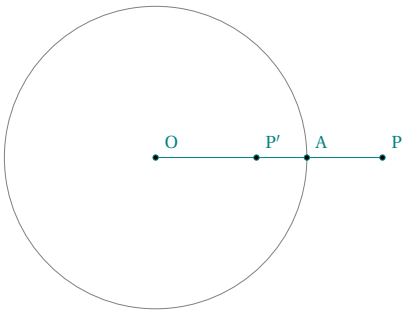

```
\begin{tikzpicture}
 \tkzDefPoint(0,0){O}
 \tkzDefPoint(0:2){A}
 \tkzDefPoint(40:2){B}
 \tkzDefPoint(20:4){C}
 \tkzDrawLine(O,A)
 \tkzDefPointBy[rotation with nodes%
             =center O from A to B](C)
 \tkzGetPoint{D}
\tkzDrawPoints(A,B,C,D)
\tkzDrawCircle(O,A)
\tkzLabelPoints(A,C,D)
\tkzLabelPoints[above](B)
\end{tikzpicture}
```

11.1.9. inversion

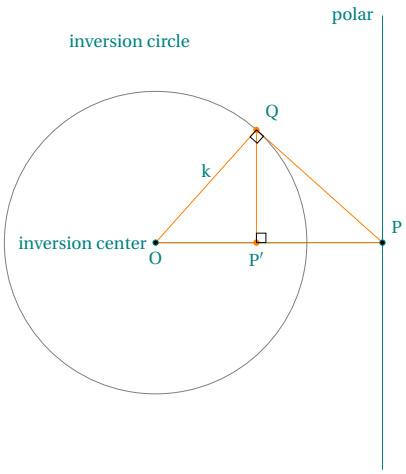
Inversion is the process of transforming points to a corresponding set of points known as their inverse points. Two points P and P′ are said to be inverses with respect to an inversion circle having inversion center O and inversion radius k if P′ is the perpendicular foot of the altitude of OQP, where Q is a point on the circle such that OQ is perpendicular to PQ.
The quantity k^2 is known as the circle power (Coxeter 1969, p. 81).
(https://mathworld.wolfram.com/Inversion.html)
Some propositions:

- The inverse of a circle (not through the center of inversion) is a circle.
- The inverse of a circle through the center of inversion is a line.
- The inverse of a line (not through the center of inversion) is a circle through the center of inversion.
- A circle orthogonal to the circle of inversion is its own inverse.
- A line through the center of inversion is its own inverse.
- Angles are preserved in inversion.

Explanation:
Directly: (center O, power $k^2 = OA^2 = OP \times OP'$)

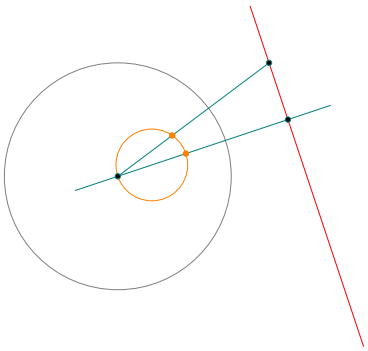



```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoints{4/0/A,6/0/P,0/0/O}
  \tkzDefPointBy[inversion = center O through A](P)
  \tkzGetPoint{P'}
  \tkzDrawSegments(O,P)
  \tkzDrawCircle(O,A)
  \tkzLabelPoints[above right,font=\scriptsize](O,A,P,P')
  \tkzDrawPoints(O,A,P,P')
\end{tikzpicture}
```



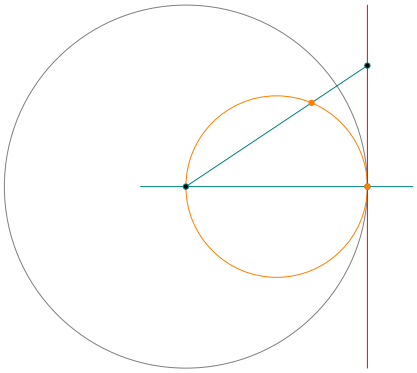

```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoints{4/0/A,6/0/P,0/0/O}
  \tkzDefLine[orthogonal=through P](O,P)
  \tkzGetPoint{L}
  \tkzDefLine[tangent from = P](O,A) \tkzGetPoints{R}{Q}
  \tkzDefPointBy[projection=onto O--A](Q) \tkzGetPoint{P'}
  \tkzDrawSegments(O,P O,A)
  \tkzDrawSegments[new](O,P O,Q P,Q Q,P')
  \tkzDrawCircle(O,A)
  \tkzDrawLines[add=1 and 0](P,L)
  \tkzLabelPoints[below,font=\scriptsize](O,P')
  \tkzLabelPoints[above right,font=\scriptsize](P,Q)
  \tkzDrawPoints(O,P) \tkzDrawPoints[new](Q,P')
  \tkzLabelSegment[above](O,Q){$k$}
  \tkzMarkRightAngles(A,P',Q P,Q,O)
  \tkzLabelCircle[above=.5cm,
      font=\scriptsize](O,A)(100){inversion circle}
  \tkzLabelPoint[left,font=\scriptsize](O){inversion center}
  \tkzLabelPoint[left,font=\scriptsize](L){polar}
\end{tikzpicture}
```

11.1.10. inversion of lines: example 1



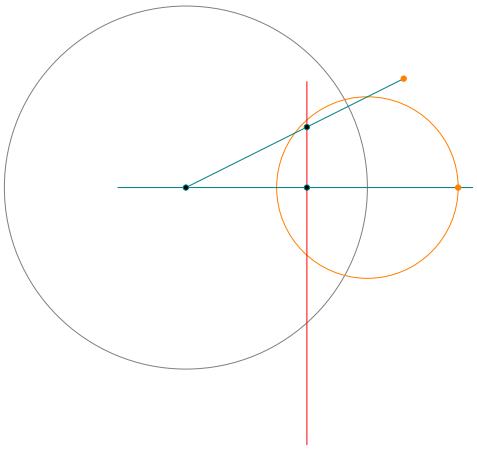
```
\begin{tikzpicture}[scale=.5]
\tkzDefPoints{0/0/O,3/0/I,4/3/P,6/-3/Q}
\tkzDrawCircle(O,I)
\tkzDefPointBy[projection= onto P--Q](O) \tkzGetPoint{A}
\tkzDefPointBy[inversion = center O through I](A)
\tkzGetPoint{A'}
\tkzDefPointBy[inversion = center O through I](P)
\tkzGetPoint{P'}
\tkzDefCircle[diameter](O,A')\tkzGetPoint{o}
\tkzDrawCircle[new](o,A')
\tkzDrawLines[add=.25 and .25,red](P,Q)
\tkzDrawLines[add=.25 and .25](O,A)
\tkzDrawSegments(O,P)
\tkzDrawPoints(A,P,O) \tkzDrawPoints[new](A',P')
\end{tikzpicture}
```

11.1.11. inversion of lines: example 2



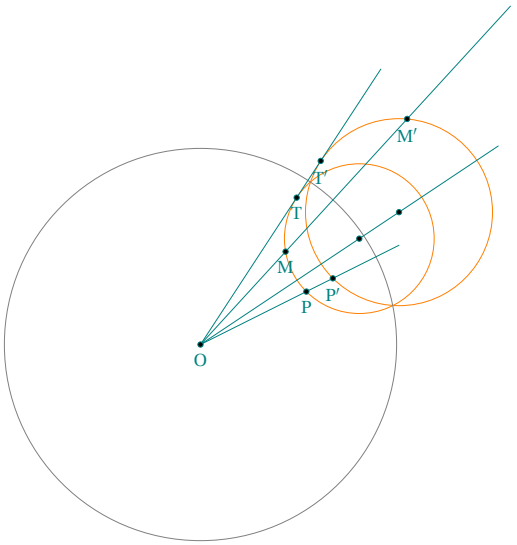
```
\begin{tikzpicture}[scale=.8]
\tkzDefPoints{0/0/O,3/0/I,3/2/P,3/-2/Q}
\tkzDrawCircle(O,I)
\tkzDefPointBy[projection= onto P--Q](O) \tkzGetPoint{A}
\tkzDefPointBy[inversion = center O through I](A)
\tkzGetPoint{A'}
\tkzDefPointBy[inversion = center O through I](P)
\tkzGetPoint{P'}
\tkzDefCircle[diameter](O,A')\tkzGetPoint{o}
\tkzDrawCircle[new](o,A')
\tkzDrawLines[add=.25 and .25,red](P,Q)
\tkzDrawLines[add=.25 and .25](O,A)
\tkzDrawSegments(O,P)
\tkzDrawPoints(A,P,O) \tkzDrawPoints[new](A',P')
\end{tikzpicture}
```

11.1.12. inversion of lines: example 3



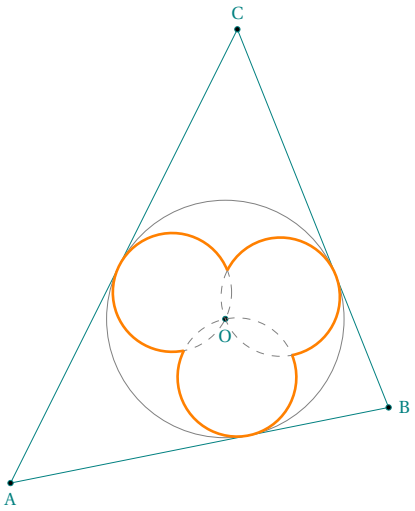
```
\begin{tikzpicture}[scale=.8]
\tkzDefPoints{0/0/O,3/0/I,2/1/P,2/-2/Q}
\tkzDrawCircle(O,I)
\tkzDefPointBy[projection= onto P--Q](O) \tkzGetPoint{A}
\tkzDefPointBy[inversion = center O through I](A)
\tkzGetPoint{A'}
\tkzDefPointBy[inversion = center O through I](P)
\tkzGetPoint{P'}
\tkzDefCircle[diameter](O,A')
\tkzDrawCircle[new](I,A')
\tkzDrawLines[add=.25 and .75,red](P,Q)
\tkzDrawLines[add=.25 and .25](O,A')
\tkzDrawSegments(O,P')
\tkzDrawPoints(A,P,O) \tkzDrawPoints[new](A',P')
\end{tikzpicture}
```

11.1.13. inversion of circle and homothety




```
\begin{tikzpicture}[scale=.7]
\tkzDefPoints{0/0/O,3/2/A,2/1/P}
\tkzDefLine[tangent from = O](A,P) \tkzGetPoints{T}{X}
\tkzDefPointsBy[homothety = center O
                ratio 1.25](A,P,T){}
\tkzInterCC(A,P)(A',P') \tkzGetPoints{C}{D}
\tkzCalcLength(A,P)
\tkzGetLength{rAP}
\tkzDefPointOnCircle[R=center A angle 190 radius \rAP]
\tkzGetPoint{M}
\tkzDefPointBy[inversion = center O through C](M)
\tkzGetPoint{M'}
\tkzDrawCircles[new](A,P A',P')
\tkzDrawCircle(O,C)
\tkzDrawLines[add=0 and .5](O,T' O,A' O,M' O,P')
\tkzDrawPoints(A,A',P,P',O,T,T',M,M')
\tkzLabelPoints(O,T,T',M,M')
\tkzLabelPoints[below](P,P')
\end{tikzpicture}
```

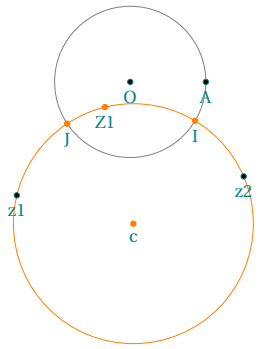
11.1.14. inversion of a triangle with respect to the incircle




```
\begin{tikzpicture}[scale=1]
\tkzDefPoints{0/0/A,5/1/B,3/6/C}
\tkzDefTriangleCenter[in](A,B,C) \tkzGetPoint{O}
\tkzDefPointBy[projection= onto A--C](O) \tkzGetPoint{b}
\tkzDefPointBy[projection= onto A--C](O) \tkzGetPoint{b}
\tkzDefPointBy[projection= onto B--C](O) \tkzGetPoint{a}
\tkzDefPointBy[projection= onto A--B](O) \tkzGetPoint{c}
\tkzDefPointsBy[inversion = center O through b](a,b,c)%
                                              {Ia,Ib,Ic}
\tkzDefMidPoint(O,Ia) \tkzGetPoint{Ja}
\tkzDefMidPoint(O,Ib) \tkzGetPoint{Jb}
\tkzDefMidPoint(O,Ic) \tkzGetPoint{Jc}
\tkzInterCC(Ja,O)(Jb,O) \tkzGetPoints{O}{x}
\tkzInterCC(Ja,O)(Jc,O) \tkzGetPoints{y}{O}
\tkzInterCC(Jb,O)(Jc,O) \tkzGetPoints{O}{z}
\tkzDrawPolygon(A,B,C)
\tkzDrawCircle(O,b)\tkzDrawPoints(A,B,C,O)
\tkzDrawCircles[dashed,gray](Ja,y Jb,x Jc,z)
\tkzDrawArc[line width=1pt,orange,delta=0](Jb,x)(z)
\tkzDrawArc[line width=1pt,orange,delta=0](Jc,z)(y)
\tkzDrawArc[line width=1pt,orange,delta=0](Ja,y)(x)
\tkzLabelPoint[below](A){$A$}\tkzLabelPoint[above](C){$C$}
\tkzLabelPoint[right](B){$B$}\tkzLabelPoint[below](O){$O$}
\end{tikzpicture}
```

11.1.15. inversion: orthogonal circle with inversion circle

The inversion circle itself, circles orthogonal to it, and lines through the inversion center are invariant under inversion. If a circle meets the circle of inversion, the points of intersection are invariant, so they also lie on the inverse circle: see I and J in the next figure.

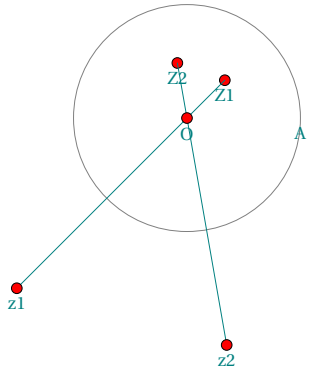



```
\begin{tikzpicture}[scale=1]
\tkzDefPoint(0,0){O}\tkzDefPoint(1,0){A}
\tkzDefPoint(-1.5,-1.5){z1}
\tkzDefPoint(1.5,-1.25){z2}
\tkzDefCircle[orthogonal through=z1 and z2](O,A)
\tkzGetPoint{c}
\tkzDrawCircle[new](c,z1)
\tkzDefPointBy[inversion =  center O through A](z1)
\tkzGetPoint{Z1}
\tkzInterCC(O,A)(c,z1) \tkzGetPoints{I}{J}
\tkzDefPointBy[inversion =  center O through A](I)
\tkzGetPoint{I'}
\tkzDrawCircle(O,A)
\tkzDrawPoints(O,A,z1,z2)
\tkzDrawPoints[new](c,Z1,I,J)
\tkzLabelPoints(O,A,z1,z2,c,Z1,I,J)
\end{tikzpicture}
```

For a more complex example, see **Pappus** 51.25.

11.1.16. inversion negative

It is an inversion followed by a symmetry with center O.




```
\begin{tikzpicture}[scale=1.5]
  \tkzDefPoints{1/0/A,0/0/O}
  \tkzDefPoint(-1.5,-1.5){z1}
  \tkzDefPoint(0.35,-2){z2}
  \tkzDefPointBy[inversion negative = center O through A](z1)
  \tkzGetPoint{Z1}
  \tkzDefPointBy[inversion negative = center O through A](z2)
  \tkzGetPoint{Z2}
  \tkzDrawCircle(O,A)
  \tkzDrawPoints[color=black, fill=red,size=4](Z1,Z2)
  \tkzDrawSegments(z1,Z1 z2,Z2)
  \tkzDrawPoints[color=black, fill=red,size=4](O,z1,z2)
  \tkzLabelPoints[font=\scriptsize](O,A,z1,z2,Z1,Z2)
\end{tikzpicture}
```

11.2. Transformation of multiple points: \tkzDefPointsBy

A variant of the previous macro, to define several images. You give the names of the images as arguments, or leave the argument empty: the names of the images are then formed from the names of the original points.

```
\tkzDefPointsBy[translation= from A to A'](B,C){}
```

The images are B′ and C′.

```
\tkzDefPointsBy[translation= from A to A'](B,C){D,E}
```

The images are D and E.

```
\tkzDefPointsBy[translation= from A to A'](B)
```

The image is B′.

\tkzDefPointsBy[⟨local options⟩](⟨list of points⟩){⟨list of points⟩}

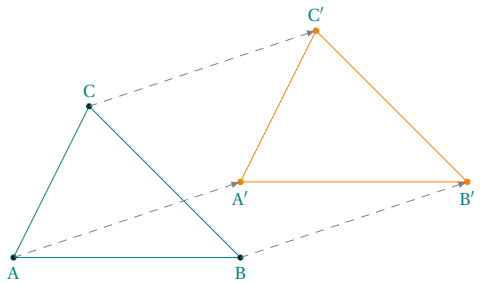
arguments	examples	
(⟨list of points⟩){⟨list of pts⟩}	(A,B){E,F}	E,F images of A, B

If the list of images is empty, the name of each image is the name of the original point followed by " ' ".

options	examples
translation = from #1 to #2	[translation=from A to B](E){}
homothety = center #1 ratio #2	[homothety=center A ratio .5](E){F}
reflection = over #1--#2	[reflection=over A--B](E){F}
symmetry = center #1	[symmetry=center A](E){F}
projection = onto #1--#2	[projection=onto A--B](E){F}
rotation = center #1 angle #2	[rotation=center angle 30](E){F}
rotation in rad = center #1 angle #2	for instance angle pi/3
rotation with nodes = center #1 from #2 to #3	[center O from A to B](E){F}
inversion = center #1 through #2	[inversion = center O through A](E){F}
inversion negative = center #1 through #2	...

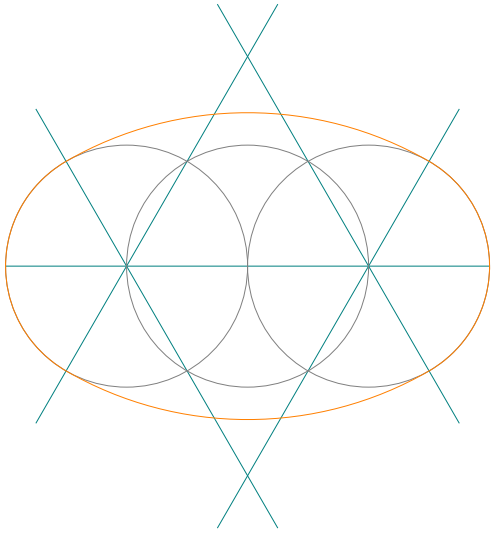
The points are only defined, not drawn.

11.2.1. translation of multiple points




```
\begin{tikzpicture}[>=latex]
 \tkzDefPoints{0/0/A,3/0/B,3/1/A',1/2/C}
 \tkzDefPointsBy[translation= from A to A'](B,C){}
 \tkzDrawPolygon(A,B,C)
 \tkzDrawPolygon[new](A',B',C')
 \tkzDrawPoints(A,B,C)
 \tkzDrawPoints[new](A',B',C')
 \tkzLabelPoints(A,B,A',B')
 \tkzLabelPoints[above](C,C')
 \tkzDrawSegments[color = gray,->,
                  style=dashed](A,A' B,B' C,C')
\end{tikzpicture}
```

11.2.2. symmetry of multiple points: an oval



```
\begin{tikzpicture}[scale=0.4]
  \tkzDefPoint(-4,0){I}
  \tkzDefPoint(4,0){J}
  \tkzDefPoint(0,0){O}
  \tkzInterCC(J,O)(O,J) \tkzGetPoints{L}{H}
  \tkzInterCC(I,O)(O,I) \tkzGetPoints{K}{G}
  \tkzInterLL(I,K)(J,H) \tkzGetPoint{M}
  \tkzInterLL(I,G)(J,L) \tkzGetPoint{N}
  \tkzDefPointsBy[symmetry=center J](L,H){D,E}
  \tkzDefPointsBy[symmetry=center I](G,K){C,F}
  \begin{scope}[line style/.style = {very thin,teal}]
    \tkzDrawLines[add=1.5 and 1.5](I,K I,G J,H J,L)
    \tkzDrawLines[add=.5 and .5](I,J)
    \tkzDrawCircles(O,I I,O J,O)
    \tkzDrawArc[delta=0,orange](N,D)(C)
    \tkzDrawArc[delta=0,orange](M,F)(E)
    \tkzDrawArc[delta=0,orange](J,E)(D)
    \tkzDrawArc[delta=0,orange](I,C)(F)
  \end{scope}
\end{tikzpicture}
```

12. Defining points using a vector

12.1. \tkzDefPointWith

Points that meet certain vector conditions are created with **\tkzDefPointWith**. The general principle is as follows: two points, i.e. a vector, are passed as arguments. Depending on the option, the new point forms with the first point (with some exceptions) a vector collinear or orthogonal to the first vector. Its length is either proportional to the length of the first vector, or proportional to the unit. Since this point is often only used temporarily, it does not have to be named immediately: the result is in **tkzPointResult**, and the macro **\tkzGetPoint** retrieves it under another name.
Some options set the distance between the given point and the point obtained. In the general case, this distance is the distance between the two points given as arguments; with the options of the **normed** type, it is 1 cm. The option K then gives multiples of this distance.

\tkzDefPointWith(⟨pt1,pt2⟩)

This macro defines a point meeting vector conditions.

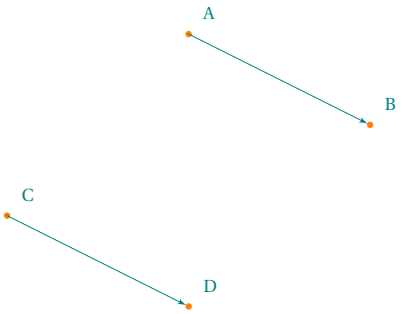
arguments	definition	explanation
(pt1,pt2)	point couple	the result is a point in **tkzPointResult**

In what follows, it is assumed that the point is recovered by **\tkzGetPoint{C}**

options	example	explanation
orthogonal	[orthogonal](A,B)	$AC = AB$ and $\overrightarrow{AC} \perp \overrightarrow{AB}$
orthogonal normed	[orthogonal normed](A,B)	$AC = 1$ and $\overrightarrow{AC} \perp \overrightarrow{AB}$
linear	[linear](A,B)	$\overrightarrow{AC} = K \times \overrightarrow{AB}$
linear normed	[linear normed](A,B)	$AC = K$ and $\overrightarrow{AC} = k \times \overrightarrow{AB}$
colinear= at #1	[colinear= at C](A,B)	$\overrightarrow{CD} = \overrightarrow{AB}$
colinear normed= at #1	[colinear normed= at C](A,B)	$\overrightarrow{CD} = \overrightarrow{AB}$
K	[linear](A,B),K=2	$\overrightarrow{AC} = 2 \times \overrightarrow{AB}$

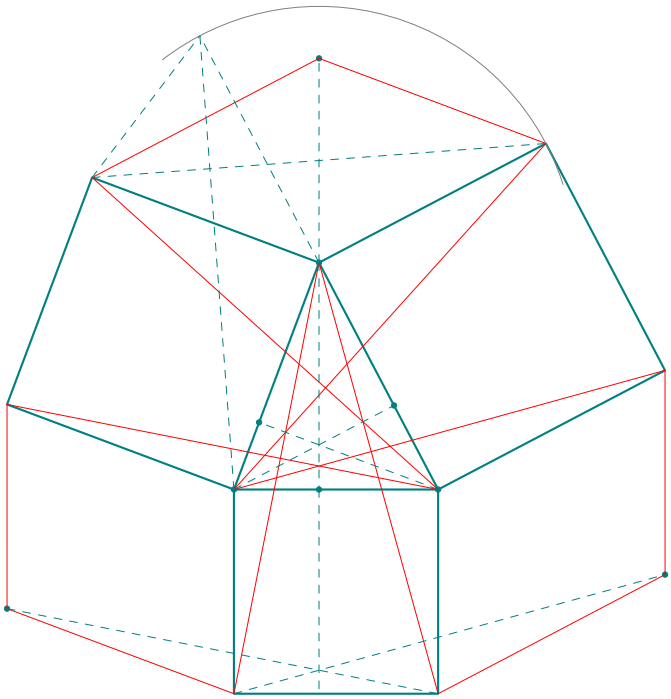
12.1.1. Option colinear at, simple example

$(\overrightarrow{AB} = \overrightarrow{CD})$




```
\begin{tikzpicture}[scale=1.2,
   vect/.style={->,shorten >=1pt,>=latex'}]
  \tkzDefPoint(2,3){A}   \tkzDefPoint(4,2){B}
  \tkzDefPoint(0,1){C}
  \tkzDefPointWith[colinear=at C](A,B)
  \tkzGetPoint{D}
  \tkzDrawPoints[new](A,B,C,D)
  \tkzLabelPoints[above right=3pt](A,B,C,D)
  \tkzDrawSegments[vect](A,B C,D)
\end{tikzpicture}
```

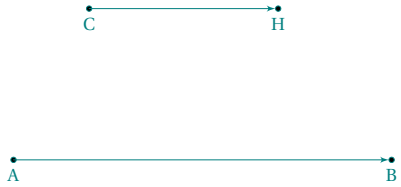
12.1.2. Option colinear at, complex example



```
\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/0/B,3.6/0/C,1.5/4/A}
\tkzDefSpcTriangle[ortho](A,B,C){Ha,Hb,Hc}
\tkzDefTriangleCenter[ortho](A,B,C) \tkzGetPoint{H}
\tkzDefSquare(A,C) \tkzGetPoints{R}{S}
\tkzDefSquare(B,A) \tkzGetPoints{M}{N}
\tkzDefSquare(C,B) \tkzGetPoints{P}{Q}
\tkzDefPointWith[colinear= at M](A,S) \tkzGetPoint{A'}
\tkzDefPointWith[colinear= at P](B,N) \tkzGetPoint{B'}
\tkzDefPointWith[colinear= at Q](C,R) \tkzGetPoint{C'}
\tkzDefPointBy[projection=onto P--Q](Ha) \tkzGetPoint{Pa}
\tkzDrawPolygon[teal,thick](A,C,R,S)\tkzDrawPolygon[teal,thick](A,B,N,M)
\tkzDrawPolygon[teal,thick](C,B,P,Q)
\tkzDrawPoints[teal,size=2](A,B,C,Ha,Hb,Hc,A',B',C')
\tkzDrawSegments[ultra thin,red](M,A' A',S P,B' B',N Q,C' C',R B,S C,M C,N B,R A,P A,Q)
\tkzDrawSegments[ultra thin,teal, dashed](A,Ha B,Hb C,Hc)
\tkzDefPointBy[rotation=center A angle 90](S) \tkzGetPoint{S'}
\tkzDrawSegments[ultra thin,teal,dashed](B,S' A,S' A,A' M,S' B',Q P,C' M,S Ha,Pa)
\tkzDrawArc(A,S)(S')
\end{tikzpicture}
```

12.1.3. Option colinear at: how to use K

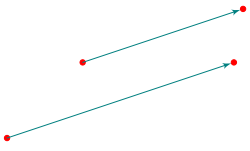
How to use K




```
\begin{tikzpicture}[vect/.style={->,
                shorten >=1pt,>=latex'}]
  \tkzDefPoints{0/0/A,5/0/B,1/2/C}
  \tkzDefPointWith[colinear=at C](A,B)
  \tkzGetPoint{G}
  \tkzDefPointWith[colinear=at C, K=0.5](A,B)
  \tkzGetPoint{H}
  \tkzLabelPoints(A,B,C,G,H)
  \tkzDrawPoints(A,B,C,G,H)
  \tkzDrawSegments[vect](A,B C,H)
\end{tikzpicture}
```

12.1.4. Option colinear at with K = $\sqrt{2}/2$

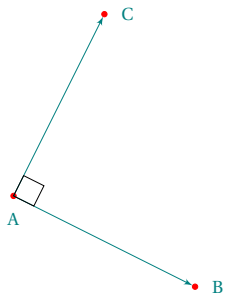
With K = $\frac{\sqrt{2}}{2}$



```
\begin{tikzpicture}[vect/.style={->,
            shorten >=1pt,>=latex'}]
 \tkzDefPoints{1/1/A,4/2/B,2/2/C}
 \tkzDefPointWith[colinear=at C,K=sqrt(2)/2](A,B)
 \tkzGetPoint{D}
 \tkzDrawPoints[color=red](A,B,C,D)
 \tkzDrawSegments[vect](A,B C,D)
\end{tikzpicture}
```

12.1.5. Option orthogonal with K = 1

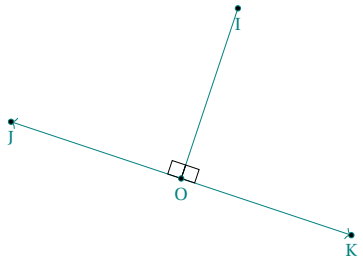
AB=AC since K = 1.




```
\begin{tikzpicture}[scale=1.2,
  vect/.style={->,shorten >=1pt,>=latex'}]
  \tkzDefPoints{2/3/A,4/2/B}
   \tkzDefPointWith[orthogonal,K=1](A,B)
     \tkzGetPoint{C}
   \tkzDrawPoints[color=red](A,B,C)
   \tkzLabelPoints[right=3pt](B,C)
   \tkzLabelPoints[below=3pt](A)
   \tkzDrawSegments[vect](A,B A,C)
   \tkzMarkRightAngle(B,A,C)
\end{tikzpicture}
```

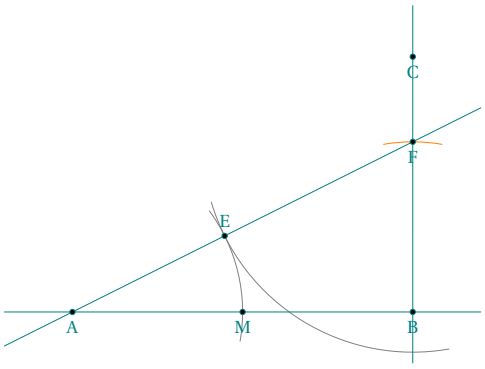
12.1.6. Option orthogonal with K = −1

With K = −1 OK=OI since |K| = 1 then OI=OJ=OK.



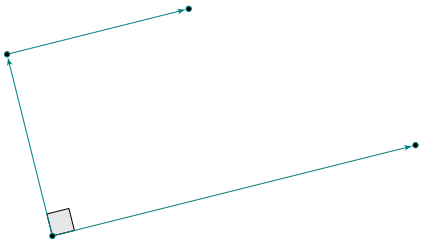

```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoints{1/2/O,2/5/I}
  \tkzDefPointWith[orthogonal](O,I)
  \tkzGetPoint{J}
  \tkzDefPointWith[orthogonal,K=-1](O,I)
  \tkzGetPoint{K}
  \tkzDrawSegment(O,I)
  \tkzDrawSegments[->](O,J O,K)
  \tkzMarkRightAngles(I,O,J I,O,K)
  \tkzDrawPoints(O,I,J,K)
  \tkzLabelPoints(O,I,J,K)
\end{tikzpicture}
```

12.1.7. Option orthogonal: a more complex example




```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoints{0/0/A,6/0/B}
  \tkzDefMidPoint(A,B)
    \tkzGetPoint{I}
  \tkzDefPointWith[orthogonal,K=-.75](B,A)
  \tkzGetPoint{C}
  \tkzInterLC(B,C)(B,I)
     \tkzGetPoints{D}{F}
  \tkzDuplicateSegment(B,F)(A,F)
  \tkzGetPoint{E}
  \tkzDrawArc[delta=10](F,E)(B)
  \tkzInterLC(A,B)(A,E)
    \tkzGetPoints{N}{M}
  \tkzDrawArc[delta=10](A,M)(E)
  \tkzDrawLines(A,B B,C A,F)
  \tkzCompass(B,F)
  \tkzDrawPoints(A,B,C,F,M,E)
  \tkzLabelPoints(A,B,C,F,M)
  \tkzLabelPoints[above](E)
\end{tikzpicture}
```

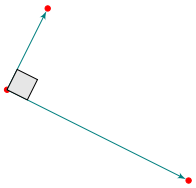
12.1.8. Options colinear and orthogonal



```
\begin{tikzpicture}[scale=1.2,
  vect/.style={->,shorten >=1pt,>=latex'}]
  \tkzDefPoints{2/1/A,6/2/B}
  \tkzDefPointWith[orthogonal,K=.5](A,B)
  \tkzGetPoint{C}
  \tkzDefPointWith[colinear=at C,K=.5](A,B)
  \tkzGetPoint{D}
  \tkzMarkRightAngle[fill=gray!20](B,A,C)
  \tkzDrawSegments[vect](A,B A,C C,D)
  \tkzDrawPoints(A,...,D)
\end{tikzpicture}
```

12.1.9. Option orthogonal normed

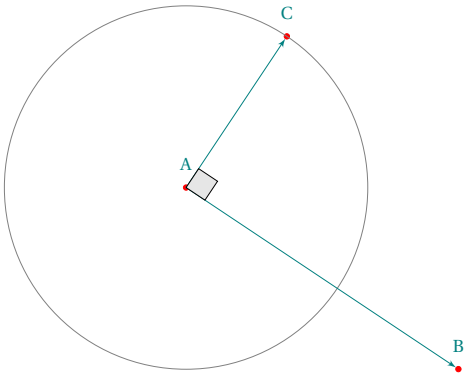
K = 1 AC = 1.



```
\begin{tikzpicture}[scale=1.2,
  vect/.style={->,shorten >=1pt,>=latex'}]
  \tkzDefPoints{2/3/A,4/2/B}
  \tkzDefPointWith[orthogonal normed](A,B)
  \tkzGetPoint{C}
  \tkzDrawPoints[color=red](A,B,C)
  \tkzDrawSegments[vect](A,B A,C)
  \tkzMarkRightAngle[fill=gray!20](B,A,C)
\end{tikzpicture}
```

12.1.10. Option orthogonal normed and K=2

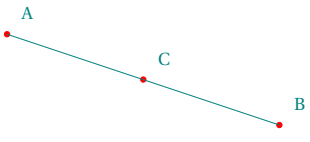
K = 2 therefore AC = 2.




```
\begin{tikzpicture}[scale=1.2,
   vect/.style={->,shorten >=1pt,>=latex'}]
  \tkzDefPoints{2/3/A,5/1/B}
  \tkzDefPointWith[orthogonal normed,K=2](A,B)
  \tkzGetPoint{C}
  \tkzDrawPoints[color=red](A,B,C)
  \tkzDefCircle[R](A,2) \tkzGetPoint{a}
  \tkzDrawCircle(A,a)
  \tkzDrawSegments[vect](A,B A,C)
  \tkzMarkRightAngle[fill=gray!20](B,A,C)
  \tkzLabelPoints[above=3pt](A,B,C)
\end{tikzpicture}
```

12.1.11. Option linear

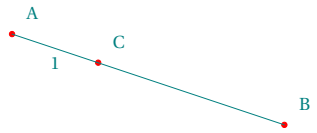
Here K = 0.5.
This amounts to applying a homothety or a multiplication of a vector by a real. Here is the midpoint of [AB].




```
\begin{tikzpicture}[scale=1.2]
  \tkzDefPoints{1/3/A,4/2/B}
  \tkzDefPointWith[linear,K=0.5](A,B)
  \tkzGetPoint{C}
  \tkzDrawPoints[color=red](A,B,C)
  \tkzDrawSegment(A,B)
  \tkzLabelPoints[above right=3pt](A,B,C)
\end{tikzpicture}
```

12.1.12. Option linear normed

In the following example AC = 1 and C belongs to (AB).




```
\begin{tikzpicture}[scale=1.2]
 \tkzDefPoints{1/3/A,4/2/B}
 \tkzDefPointWith[linear normed](A,B)
 \tkzGetPoint{C}
 \tkzDrawPoints[color=red](A,B,C)
 \tkzDrawSegment(A,B)
 \tkzLabelSegment(A,C){$1$}
 \tkzLabelPoints[above right=3pt](A,B,C)
\end{tikzpicture}
```

12.2. \tkzGetVectxy

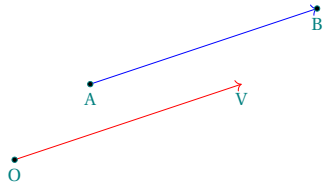
Retrieving the coordinates of a vector.

\tkzGetVectxy(⟨A,B⟩){⟨text⟩}

Gives the coordinates of a vector.

arguments	example	explanation
(point){name of macro}	\tkzGetVectxy(A,B){V}	\Vx,\Vy: coordinates of $\overrightarrow{AB}$

12.2.1. Coordinate transfer with \tkzGetVectxy




```
\begin{tikzpicture}
 \tkzDefPoints{0/0/O,1/1/A,4/2/B}
 \tkzGetVectxy(A,B){v}
 \tkzDefPoint(\vx,\vy){V}
 \tkzDrawSegment[->,color=red](O,V)
 \tkzDrawSegment[->,color=blue](A,B)
 \tkzDrawPoints(A,B,O)
 \tkzLabelPoints(A,B,O,V)
\end{tikzpicture}
```

13. Random points

A random point can be placed:

1. in a rectangle;
2. on a segment;
3. on a straight line;
4. on a circle;
5. in a disc.

13.1. Obtaining random points

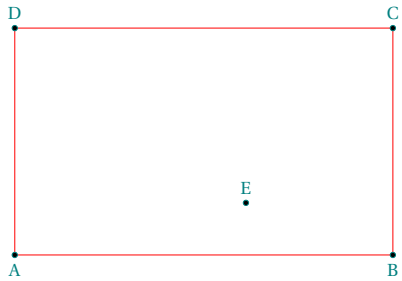
This macro replaces the old \tkzGetRandPointOn.

\tkzDefRandPointOn[⟨local options⟩]

The result is a point with a random position, which you can name with the macro **\tkzGetPoint**. You can use **tkzPointResult** directly if you don't need to keep the result.

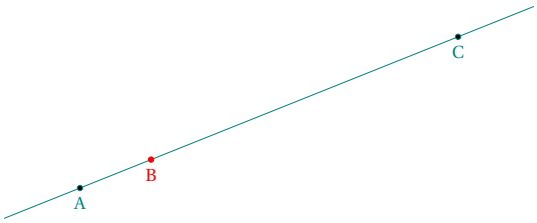
options	default	definition
rectangle=pt1 and pt2		[rectangle=A and B]
segment= pt1--pt2		[segment=A--B]
line=pt1--pt2		[line=A--B]
circle =center pt1 radius dim		[circle = center A radius 2]
circle through=center pt1 through pt2		[circle through= center A through B]
disk through=center pt1 through pt2		[disk through=center A through B]

13.1.1. Random point in a rectangle



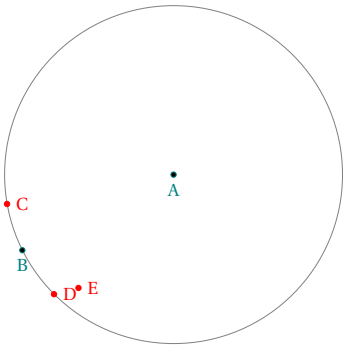

```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,5/3/C}
  \tkzDefRandPointOn[rectangle = A and C]
  \tkzGetPoint{E}
  \tkzDefRectangle(A,C)\tkzGetPoints{B}{D}
  \tkzDrawPolygon[red](A,...,D)
  \tkzDrawPoints(A,...,E)
  \tkzLabelPoints(A,B)
  \tkzLabelPoints[above](C,D,E)
\end{tikzpicture}
```

13.1.2. Random point on a segment or a line




```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,5/2/C}
  \tkzDefRandPointOn[segment = A--C]\tkzGetPoint{B}
  \tkzDrawLine(A,C)
  \tkzDrawPoints(A,C) \tkzDrawPoint[red](B)
  \tkzLabelPoints(A,C) \tkzLabelPoints[red](B)
\end{tikzpicture}
```

13.1.3. Random point on a circle or a disk




```
\begin{tikzpicture}
\tkzDefPoints{3/2/A,1/1/B}
\tkzCalcLength(A,B) \tkzGetLength{rAB}
\tkzDefRandPointOn[circle = center A radius \rAB]
\tkzGetPoint{C}
\tkzDefRandPointOn[circle through= center A through B]
\tkzGetPoint{D}
\tkzDefRandPointOn[disk through=center A through B]
\tkzGetPoint{E}
\tkzDrawCircle(A,B)
\tkzDrawPoints(A,B)
\tkzLabelPoints(A,B)
\tkzDrawPoints[red](C,D,E)
\tkzLabelPoints[red,right](C,D,E)
\end{tikzpicture}
```

Part III.

Lines, triangles and polygons

14. Straight lines

Drawing straight lines is of course essential, but first we must be able to define particular lines such as mediators (perpendicular bisectors), bisectors, parallels and perpendiculars. The principle is to determine two points of the line.

14.1. Definition of straight lines

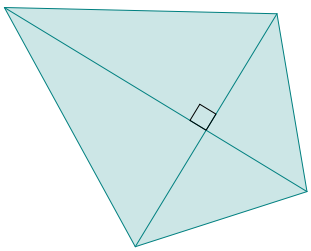
\tkzDefLine[⟨local options⟩](⟨pt1,pt2⟩) or (⟨pt1,pt2,pt3⟩)

The argument is a list of two or three points. Depending on the case, the macro defines one or two points needed to obtain the line; retrieve them with **\tkzGetPoint** or **\tkzGetPoints**. I use the term "mediator" for the perpendicular bisector of a segment.

arguments	example	explanation
(⟨pt1,pt2⟩)	[mediator](⟨A,B⟩)	mediator of the segment [A,B]
(⟨pt1,pt2,pt3⟩)	[bisector](⟨A,B,C⟩)	bisector of $\widehat{ABC}$
(⟨pt1,pt2,pt3⟩)	[altitude](⟨A,B,C⟩)	altitude from B
(⟨pt1⟩)	[tangent at=A](⟨O⟩)	tangent at A on the circle center O
(⟨pt1,pt2⟩)	[tangent from=A](⟨O,B⟩)	circle center O through B

options	default	definition
mediator		perpendicular bisector of a line segment
perpendicular=through…	mediator	perpendicular to a line passing through a point
orthogonal=through…	mediator	see above
parallel=through…	mediator	parallel to a line passing through a point
bisector	mediator	bisector of an angle defined by three points
bisector out	mediator	exterior angle bisector
symmedian	mediator	symmedian from a vertex
altitude	mediator	altitude from avertex
euler	mediator	euler line of a triangle
tangent at	mediator	tangent at a point of a circle
tangent from	mediator	tangent from an exterior point
K	1	coefficient for the perpendicular line
normed	false	normalizes the created segment

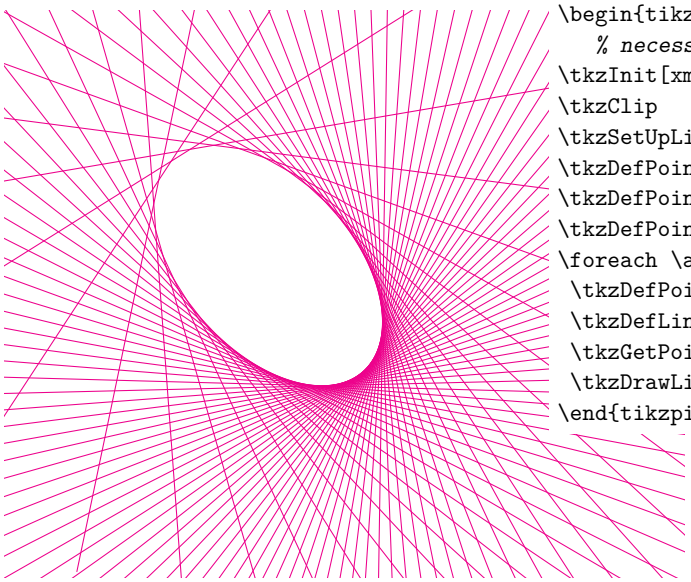
14.1.1. With mediator



```
\begin{tikzpicture}[rotate=25]
 \tkzDefPoints{-2/0/A,1/2/B}
 \tkzDefLine[mediator](A,B)          \tkzGetPoints{C}{D}
 \tkzDefPointWith[linear,K=.75](C,D) \tkzGetPoint{D}
 \tkzDefMidPoint(A,B)                \tkzGetPoint{I}
 \tkzFillPolygon[color=teal!20](A,C,B,D)
 \tkzDrawSegments(A,B C,D)
 \tkzMarkRightAngle(B,I,C)
 \tkzDrawSegments(D,B D,A)
 \tkzDrawSegments(C,B C,A)
\end{tikzpicture}
```

14.1.2. An envelope with option mediator

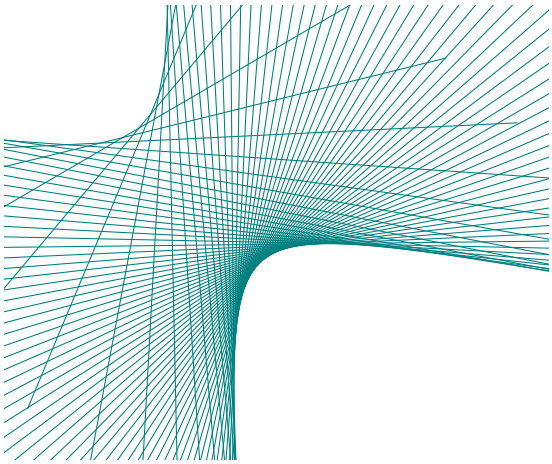
Based on a figure by O. Reboux, made with pst-eucl by D. Rodriguez.



```
\begin{tikzpicture}[scale=.75]
   % necessary
\tkzInit[xmin=-6,ymin=-4,xmax=6,ymax=6]
\tkzClip
\tkzSetUpLine[thin,color=magenta]
\tkzDefPoint(0,0){O}
\tkzDefPoint(132:4){A}
\tkzDefPoint(5,0){B}
\foreach \ang in {5,10,...,360}{%
 \tkzDefPoint(\ang:5){M}
 \tkzDefLine[mediator](A,M)
 \tkzGetPoints{x}{y}
 \tkzDrawLine[add= 3 and 3](x,y)}
\end{tikzpicture}
```

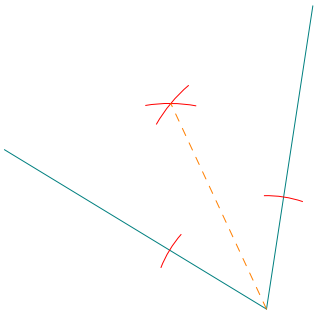
14.1.3. A parabola with option mediator

Based on a figure by O. Reboux, made with pst-eucl by D. Rodriguez. It is not necessary to name the two points that define the mediator.



```
\begin{tikzpicture}[scale=.6]
\tkzInit[xmin=-6,ymin=-4,xmax=6,ymax=6]
\tkzClip
\tkzSetUpLine[thin,color=teal]
\tkzDefPoint(0,0){O}
\tkzDefPoint(132:5){A}
\tkzDefPoint(4,0){B}
\foreach \ang in {5,10,...,360}{%
 \tkzDefPoint(\ang:4){M}
 \tkzDefLine[mediator](A,M)
 \tkzGetPoints{x}{y}
 \tkzDrawLine[add= 3 and 3](x,y)}
\end{tikzpicture}
```

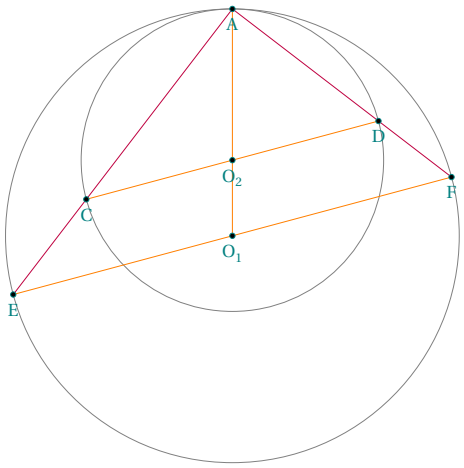
14.1.4. With options bisector and normed



```
\begin{tikzpicture}[rotate=25,scale=.75]
 \tkzDefPoints{0/0/C, 2/-3/A, 4/0/B}
 \tkzDefLine[bisector,normed](B,A,C) \tkzGetPoint{a}
 \tkzDrawLines[add= 0 and .5](A,B A,C)
 \tkzShowLine[bisector,gap=4,size=2,color=red](B,A,C)
 \tkzDrawLines[new,dashed,add= 0 and 3](A,a)
\end{tikzpicture}
```

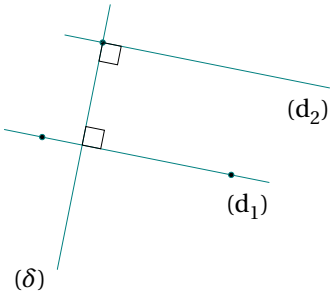
14.1.5. With option parallel=through

Archimedes' Book of Lemmas, Proposition 1.



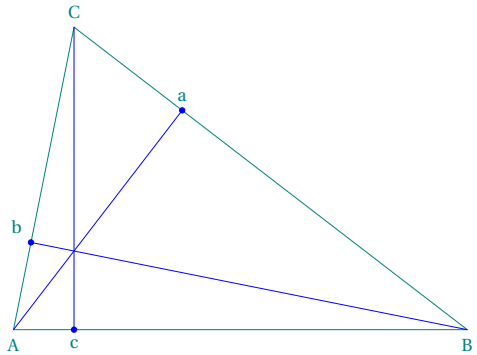

```
\begin{tikzpicture}
  \tkzDefPoints{0/0/O_1,0/1/O_2,0/3/A}
  \tkzDefPoint(15:3){F}
  \tkzDefPointBy[symmetry=center O_1](F)
  \tkzGetPoint{E}
  \tkzDefLine[parallel=through O_2](E,F)
  \tkzGetPoint{x}
  \tkzInterLC(x,O_2)(O_2,A) \tkzGetPoints{D}{C}
  \tkzDrawCircles(O_1,A O_2,A)
  \tkzDrawSegments[new](O_1,A E,F C,D)
  \tkzDrawSegments[purple](A,E A,F)
  \tkzDrawPoints(A,O_1,O_2,E,F,C,D)
  \tkzLabelPoints(A,O_1,O_2,E,F,C,D)
\end{tikzpicture}
```

14.1.6. With option orthogonal and parallel



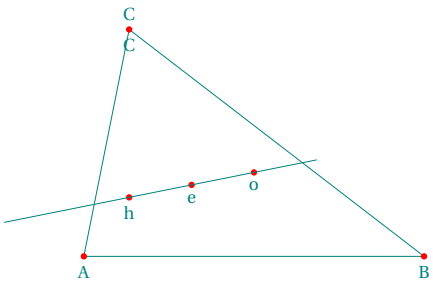

```
\begin{tikzpicture}
   \tkzDefPoints{-1.5/-0.25/A,1/-0.75/B,-0.7/1/C}
   \tkzDrawLine(A,B)
   \tkzLabelLine[pos=1.25,below left](A,B){$(d_1)$}
   \tkzDrawPoints(A,B,C)
   \tkzDefLine[orthogonal=through C](B,A) \tkzGetPoint{c}
   \tkzDrawLine(C,c)
   \tkzLabelLine[pos=1.25,left](C,c){$(\delta)$}
   \tkzInterLL(A,B)(C,c) \tkzGetPoint{I}
   \tkzMarkRightAngle(C,I,B)
   \tkzDefLine[parallel=through C](A,B) \tkzGetPoint{c'}
   \tkzDrawLine(C,c')
   \tkzLabelLine[pos=1.25,below left](C,c'){$(d_2)$}
   \tkzMarkRightAngle(I,C,c')
\end{tikzpicture}
```

14.1.7. With option altitude



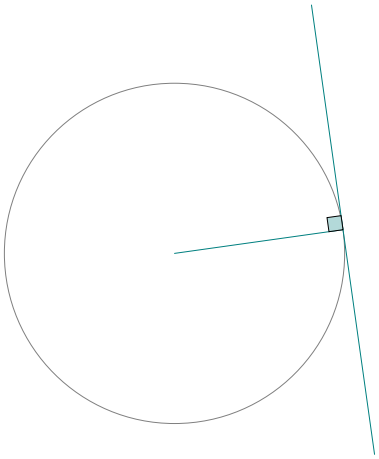

```
\begin{tikzpicture}
\tkzDefPoints{0/0/A,6/0/B,0.8/4/C}
\tkzDefLine[altitude](A,B,C)      \tkzGetPoint{b}
\tkzDefLine[altitude](B,C,A)      \tkzGetPoint{c}
\tkzDefLine[altitude](B,A,C)      \tkzGetPoint{a}
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints[blue](a,b,c)
\tkzDrawSegments[blue](A,a B,b C,c)
\tkzLabelPoints(A,B,c)
\tkzLabelPoints[above](C,a)
\tkzLabelPoints[above left](b)
\end{tikzpicture}
```

14.1.8. With option euler




```
\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/0/A,6/0/B,0.8/4/C}
\tkzDefLine[euler](A,B,C)
\tkzGetPoints{h}{e}
\tkzDefTriangleCenter[circum](A,B,C)
\tkzGetPoint{o}
\tkzDrawPolygon[teal](A,B,C)
\tkzDrawPoints[red](A,B,C,h,e,o)
\tkzDrawLine[add= 2 and 2](h,e)
\tkzLabelPoints(A,B,C,h,e,o)
\tkzLabelPoints[above](C)
\end{tikzpicture}
```

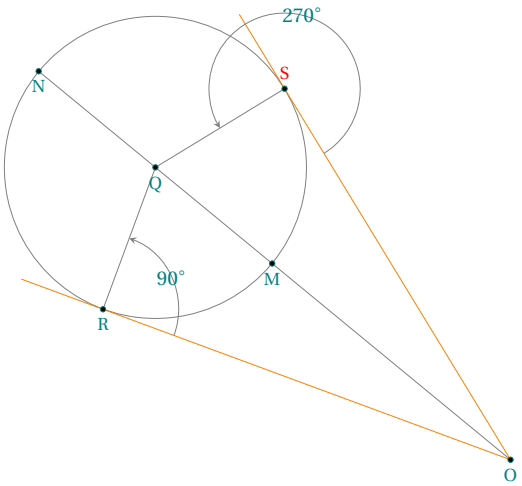
14.1.9. Tangent passing through a point on the circle tangent at



```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(6,6){E}
  \tkzDefRandPointOn[circle=center O radius 3]
  \tkzGetPoint{A}
  \tkzDrawSegment(O,A)
  \tkzDrawCircle(O,A)
  \tkzDefLine[tangent at=A](O)
  \tkzGetPoint{h}
  \tkzDrawLine[add = 4 and 3](A,h)
  \tkzMarkRightAngle[fill=teal!30](O,A,h)
\end{tikzpicture}
```

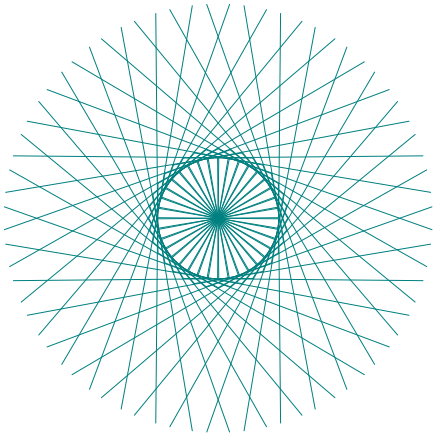
14.1.10. Choice of contact point with tangents passing through an external point option tangent from

The tangent is not drawn. With the option **at**, a point of the tangent is given by **tkzPointResult**. With the option **from**, you get two points of the circle, **tkzFirstPointResult** and **tkzSecondPointResult**. You can choose between them by comparing the angles formed by the outer point, the contact point and the center: the two angles have opposite orientations, and the counterclockwise one corresponds to **tkzFirstPointResult**.



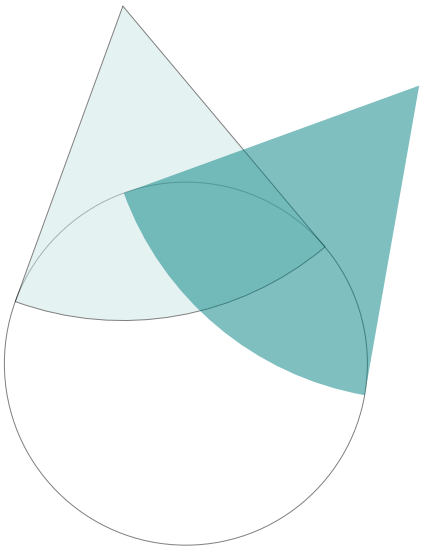

```
\begin{tikzpicture}[scale=1,rotate=-30]
\tkzDefPoints{0/0/Q,0/2/A,6/-1/O}
\tkzDefLine[tangent from = O](Q,A)
\tkzGetPoints{R}{S}
\tkzInterLC[near](O,Q)(Q,A)
\tkzGetPoints{M}{N}
\tkzDrawCircle(Q,M)
\tkzDrawSegments[new,add = 0 and .2](O,R O,S)
\tkzDrawSegments[gray](N,O R,Q S,Q)
\tkzDrawPoints(O,Q,R,S,M,N)
\tkzMarkAngle[gray,-stealth,size=1](O,R,Q)
\tkzFindAngle(O,R,Q)   \tkzGetAngle{an}
\tkzLabelAngle(O,R,Q){%
    $\pgfmathprintnumber{\an}^\circ$}
\tkzMarkAngle[gray,-stealth,size=1](O,S,Q)
\tkzFindAngle(O,S,Q)   \tkzGetAngle{an}
\tkzLabelAngle(O,S,Q){%
    $\pgfmathprintnumber{\an}^\circ$}
\tkzLabelPoints(Q,O,M,N,R)
\tkzLabelPoints[above,text=red](S)
\end{tikzpicture}
```

14.1.11. Example of tangents passing through an external point



```
\begin{tikzpicture}[scale=.8]
\tkzDefPoints{0/0/c,1/0/d,3/0/a0}
\def\tkzRadius{1}
\tkzDrawCircle(c,d)
 \foreach \an in {0,10,...,350}{
  \tkzDefPointBy[rotation=center c angle \an](a0)
  \tkzGetPoint{a}
  \tkzDefLine[tangent from = a](c,d)
  \tkzGetPoints{e}{f}
  \tkzDrawLines(a,f a,e)
  \tkzDrawSegments(c,e c,f)}
\end{tikzpicture}
```

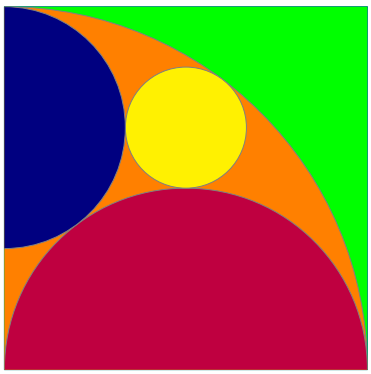
14.1.12. Example of Andrew Mertz



```
\begin{tikzpicture}[scale=.6]
 \tkzDefPoint(100:8){A}\tkzDefPoint(50:8){B}
 \tkzDefPoint(0,0){C} \tkzDefPoint(0,-4){R}
 \tkzDrawCircle(C,R)
 \tkzDefLine[tangent from = A](C,R)  \tkzGetPoints{D}{E}
\tkzDefLine[tangent from = B](C,R)  \tkzGetPoints{F}{G}
 \tkzDrawSector[fill=teal!20,opacity=0.5](A,E)(D)
 \tkzFillSector[color=teal,opacity=0.5](B,G)(F)
 \end{tikzpicture}
```

http://www.texample.net/tikz/examples/

14.1.13. Drawing a tangent option **tangent from**



```
\begin{tikzpicture}[scale=.6]
 \tkzDefPoint(0,0){B}
 \tkzDefPoint(0,8){A}
 \tkzDefSquare(A,B)
 \tkzGetPoints{C}{D}
 \tkzDrawPolygon(A,B,C,D)
 \tkzClipPolygon(A,B,C,D)
 \tkzDefPoint(4,8){F}
 \tkzDefPoint(4,0){E}
 \tkzDefPoint(4,4){Q}
 \tkzFillPolygon[color = green](A,B,C,D)
 \tkzDrawCircle[fill = orange](B,A)
 \tkzDrawCircle[fill = purple](E,B)
 \tkzDefLine[tangent from = B](F,A)
 \tkzInterLL(F,tkzSecondPointResult)(C,D)
 \tkzInterLL(A,tkzPointResult)(F,E)
 \tkzDrawCircle[fill = yellow](tkzPointResult,Q)
 \tkzDefPointBy[projection= onto B--A](tkzPointResult)
 \tkzDrawCircle[fill = blue!50!black](tkzPointResult,A)
\end{tikzpicture}
```

15. Triangles

15.1. Definition of triangles \tkzDefTriangle

The following macros define or construct a triangle from **at least** two points.
The following triangles can be defined:

- **two angles** determines a triangle with two angles;
- **equilateral** determines an equilateral triangle;
- **isosceles right** determines an isosceles right triangle;

- `half` determines a right triangle whose legs (the two sides of the right angle) are in the ratio 2;
- `pythagore` determines a right triangle whose sides are proportional to 3, 4 and 5;
- `school` determines a right triangle whose angles are 30, 60 and 90 degrees;
- `golden` (or `sublime`) determines the golden triangle: an isosceles triangle whose base is [AB] and whose base angles are 72 degrees;
- `euclid` determines Euclid's triangle, which has the same angles but whose side is [AB] (36 degrees at A); in previous versions the option was "euclide" with an "e";
- `gold` determines a right triangle, with the right angle at B, whose legs are in the ratio $\Phi = 1.618034$: AB/BC = Φ. This triangle comes from the golden rectangle.
- `cheops` determines a third point such that the triangle is isosceles, with sides proportional to 2, Φ and Φ.

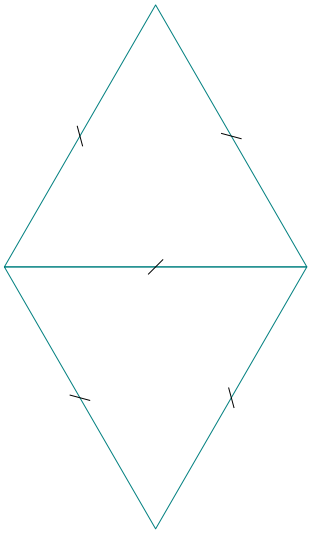
`\tkzDefTriangle[⟨local options⟩](⟨A,B⟩)`

The order of the points matters, because the triangle is constructed counterclockwise. The macro is used either with `\tkzGetPoint`, or directly with `tkzPointResult` if you don't need to keep the point.

options	default	definition
two angles= #1 and #2	no defaut	triangle knowing two angles
equilateral	equilateral	equilateral triangle
half	equilateral	B rectangle AB = 2BC AC hypothenuse
isosceles right	equilateral	isosceles right triangle
pythagore	equilateral	proportional to the pythagorean triangle 3-4-5
pythagoras	equilateral	same as above
egyptian	equilateral	same as above
school	equilateral	angles of 30, 60 and 90 degrees
gold	equilateral	B rectangle and AB/AC = Φ
euclid	equilateral	angles of 72, 72 and 36 degrees, A is the apex
golden	equilateral	angles of 72, 72 and 36 degrees, C is the apex
sublime	equilateral	angles of 72, 72 and 36 degrees, C is the apex
cheops	equilateral	AC=BC, AC and BC are proportional to 2 and Φ.
swap	false	gives the symmetric point with respect to AB

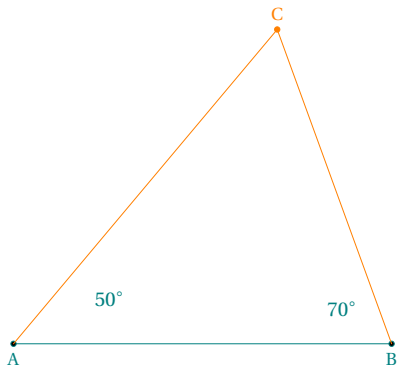
`\tkzGetPoint` stores the point; otherwise `tkzPointResult` can be used immediately.

15.1.1. Option equilateral



```
\begin{tikzpicture}
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(4,0){B}
  \tkzDefTriangle[equilateral](A,B)
  \tkzGetPoint{C}
  \tkzDrawPolygons(A,B,C)
  \tkzDefTriangle[equilateral](B,A)
  \tkzGetPoint{D}
  \tkzDrawPolygon(B,A,D)
  \tkzMarkSegments[mark=s|](A,B B,C A,C A,D B,D)
\end{tikzpicture}
```

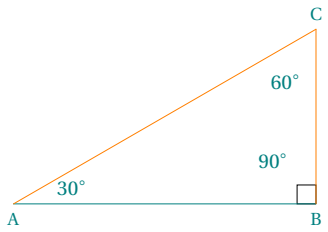
15.1.2. Option two angles




```
\begin{tikzpicture}
\tkzDefPoint(0,0){A}
\tkzDefPoint(5,0){B}
\tkzDefTriangle[two angles = 50 and 70](A,B)
\tkzGetPoint{C}
\tkzDrawSegment(A,B)
\tkzDrawPoints(A,B)
\tkzLabelPoints(A,B)
\tkzDrawSegments[new](A,C B,C)
\tkzDrawPoints[new](C)
\tkzLabelPoints[above,new](C)
\tkzLabelAngle[pos=1.4](B,A,C){$50^\circ$}
\tkzLabelAngle[pos=0.8](C,B,A){$70^\circ$}
\end{tikzpicture}
```

15.1.3. Option school

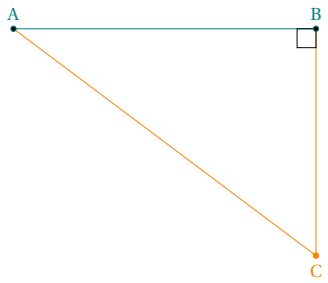
The angles are 30, 60 and 90 degrees.




```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,4/0/B}
  \tkzDefTriangle[school](A,B)
  \tkzGetPoint{C}
  \tkzMarkRightAngles(C,B,A)
  \tkzLabelAngle[pos=0.8](B,A,C){$30^\circ$}
  \tkzLabelAngle[pos=0.8](C,B,A){$90^\circ$}
  \tkzLabelAngle[pos=0.8](A,C,B){$60^\circ$}
  \tkzDrawSegments(A,B)
  \tkzDrawSegments[new](A,C B,C)
  \tkzLabelPoints(A,B)
  \tkzLabelPoints[above](C)
\end{tikzpicture}
```

15.1.4. Option pythagore

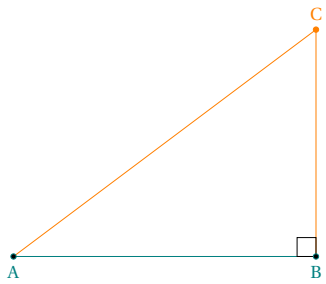
This triangle has sides whose lengths are proportional to 3, 4 and 5.




```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,4/0/B}
  \tkzDefTriangle[pythagore](A,B)
  \tkzGetPoint{C}
  \tkzDrawSegments(A,B)
  \tkzDrawSegments[new](A,C B,C)
  \tkzMarkRightAngles(A,B,C)
  \tkzDrawPoints[new](C)
  \tkzDrawPoints(A,B)
  \tkzLabelPoints[above](A,B)
  \tkzLabelPoints[new](C)
\end{tikzpicture}
```

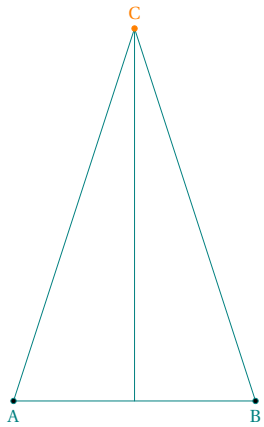
15.1.5. Option pythagore and swap

This triangle has sides whose lengths are proportional to 3, 4 and 5.




```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,4/0/B}
  \tkzDefTriangle[pythagore,swap](A,B)
  \tkzGetPoint{C}
  \tkzDrawSegments(A,B)
  \tkzDrawSegments[new](A,C B,C)
  \tkzMarkRightAngles(A,B,C)
  \tkzLabelPoint[above,new](C){$C$}
  \tkzDrawPoints[new](C)
  \tkzDrawPoints(A,B)
  \tkzLabelPoints(A,B)
\end{tikzpicture}
```

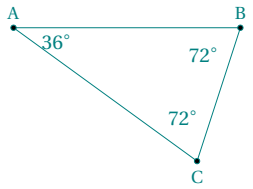
15.1.6. Option golden




```
\begin{tikzpicture}[scale=.8]
\tkzDefPoint(0,0){A} \tkzDefPoint(4,0){B}
\tkzDefTriangle[golden](A,B)\tkzGetPoint{C}
\tkzDefSpcTriangle[in,name=M](A,B,C){a,b,c}
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints(A,B)
\tkzDrawSegment(C,Mc)
\tkzDrawPoints[new](C)
\tkzLabelPoints(A,B)
\tkzLabelPoints[above,new](C)
\end{tikzpicture}
```

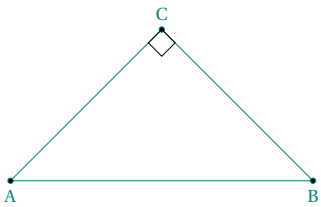
15.1.7. Option euclid

The euclid and golden triangles have the same shape, but the segment [AB] is a side in the first one and the base in the second one.



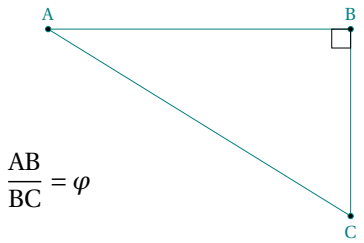

```
\begin{tikzpicture}[scale=.75]
 \tkzDefPoint(0,0){A} \tkzDefPoint(4,0){B}
 \tkzDefTriangle[euclid](A,B)\tkzGetPoint{C}
 \tkzDrawPolygon(A,B,C)
 \tkzDrawPoints(A,B,C)
 \tkzLabelPoints(C)
 \tkzLabelPoints[above](A,B)
 \tkzLabelAngle[pos=0.8](A,B,C){$72^\circ$}
 \tkzLabelAngle[pos=0.8](B,C,A){$72^\circ$}
 \tkzLabelAngle[pos=0.8](C,A,B){$36^\circ$}
\end{tikzpicture}
```

15.1.8. Option isosceles right




```
\begin{tikzpicture}
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(4,0){B}
  \tkzDefTriangle[isosceles right](A,B)
  \tkzGetPoint{C}
  \tkzDrawPolygons(A,B,C)
  \tkzDrawPoints(A,B,C)
  \tkzMarkRightAngles(A,C,B)
  \tkzLabelPoints(A,B)
  \tkzLabelPoints[above](C)
\end{tikzpicture}
```

15.1.9. Option gold




```
\begin{tikzpicture}
 \tkzDefPoints{0/0/A,4/0/B}
 \tkzDefTriangle[gold](A,B)
 \tkzGetPoint{C}
 \tkzDrawPolygon(A,B,C)
 \tkzDrawPoints(A,B,C)
 \tkzLabelPoints[above](A,B)
 \tkzLabelPoints[below](C)
 \tkzMarkRightAngle(A,B,C)
 \tkzText(0,-2){$\dfrac{AB}{BC}=\varphi$}
\end{tikzpicture}
```

15.2. Specific triangles with \tkzDefSpcTriangle

The centers of some triangles have been defined in the section on special points; here we determine the three vertices of specific triangles.

\tkzDefSpcTriangle[⟨local options⟩](⟨p1,p2,p3⟩){⟨r1,r2,r3⟩}

The order of the points is important! p1p2p3 defines a triangle; the result is a triangle whose vertices have as references **name** followed by r1, r2 and r3. If **name** is empty, the references are r1, r2 and r3.

options	default	definition
orthic	centroid	determined by endpoints of the altitudes ...
centroid or medial	centroid	intersection of the triangle's three triangle medians
in or incentral	centroid	determined with the angle bisectors
ex or excentral	centroid	determined with the excenters
extouch	centroid	formed by the points of tangency with the excircles
intouch or contact	centroid	formed by the points of tangency of the incircle each of the vertices
euler	centroid	formed by Euler points on the nine-point circle
symmedial	centroid	intersection points of the symmedians
tangential	centroid	formed by the lines tangent to the circumcircle
feuerbach	centroid	formed by the points of tangency of the nine-point ... circle with the excircles
name	empty	used to name the vertices

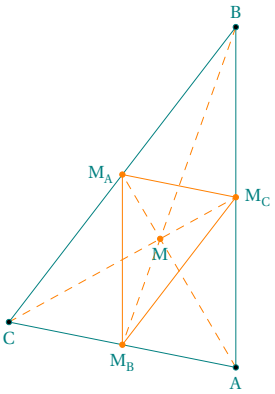
15.2.1. How to name the vertices

With `\tkzDefSpcTriangle[medial,name=M](A,B,C){_A,_B,_C}` you get three vertices named M_A, M_B and M_C.
With `\tkzDefSpcTriangle[medial](A,B,C){a,b,c}` you get three vertices named a, b and c.
With `\tkzDefSpcTriangle[medial,name=M_](A,B,C){A,B,C}` you also get three vertices named M_A, M_B and M_C.

15.2.2. Option medial or centroid

The centroid of a triangle is the point G (sometimes also denoted M) where the three medians of the triangle meet; it is therefore sometimes called the median point. The centroid always lies inside the triangle.

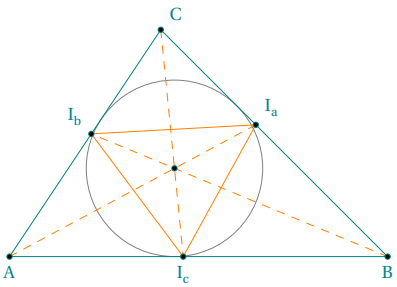
Weisstein, Eric W. "Centroid triangle" From MathWorld–A Wolfram Web Resource.
In the following example, we get the Euler circle, which passes through the points defined above.




```
\begin{tikzpicture}[rotate=90,scale=.75]
 \tkzDefPoints{0/0/A,6/0/B,0.8/4/C}
 \tkzDefTriangleCenter[centroid](A,B,C)
 \tkzGetPoint{M}
 \tkzDefSpcTriangle[medial,name=M](A,B,C){_A,_B,_C}
 \tkzDrawPolygon(A,B,C)
 \tkzDrawSegments[dashed,new](A,M_A B,M_B C,M_C)
 \tkzDrawPolygon[new](M_A,M_B,M_C)
 \tkzDrawPoints(A,B,C)
 \tkzDrawPoints[new](M,M_A,M_B,M_C)
 \tkzLabelPoints[above](B)
 \tkzLabelPoints[below](A,C,M_B)
 \tkzLabelPoints[right](M_C)
 \tkzLabelPoints[left](M_A)
 \tkzLabelPoints[font=\scriptsize](M)
\end{tikzpicture}
```

15.2.3. Option in or incentral

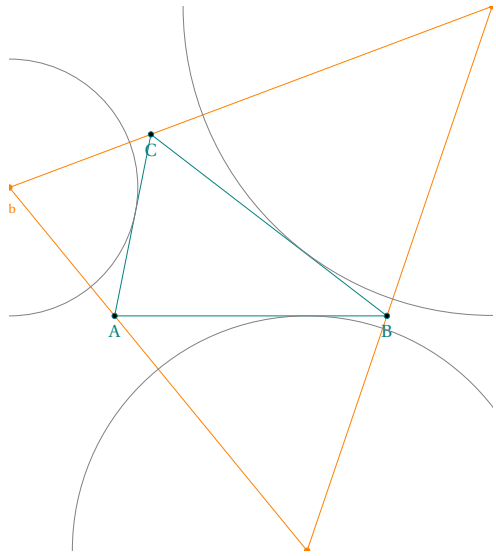
The incentral triangle is the triangle whose vertices are determined by the intersections of the reference triangle's angle bisectors with the respective opposite sides.
Weisstein, Eric W. "Incentral triangle" From MathWorld–A Wolfram Web Resource.




```
\begin{tikzpicture}[scale=1]
  \tkzDefPoints{ 0/0/A,5/0/B,2/3/C}
  \tkzDefSpcTriangle[in,name=I](A,B,C){_a,_b,_c}
  \tkzDefCircle[in](A,B,C) \tkzGetPoints{I}{a}
  \tkzDrawCircle(I,a)
  \tkzDrawPolygon(A,B,C)
  \tkzDrawPolygon[new](I_a,I_b,I_c)
  \tkzDrawSegments[dashed,new](A,I_a B,I_b C,I_c)
  \tkzDrawPoints(A,B,C,I,I_a,I_b,I_c)
  \tkzLabelPoints[below](A,B,I_c)
  \tkzLabelPoints[above left](I_b)
  \tkzLabelPoints[above right](C,I_a)
\end{tikzpicture}
```

15.2.4. Option ex or excentral

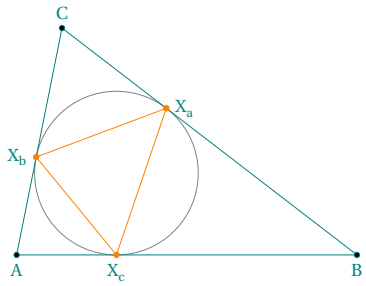
The excentral triangle of a triangle ABC is the triangle $J_aJ_bJ_c$ with vertices corresponding to the excenters of ABC.




```
\begin{tikzpicture}[scale=.6]
 \tkzDefPoints{0/0/A,6/0/B,0.8/4/C}
 \tkzDefSpcTriangle[excentral,name=J](A,B,C){_a,_b,_c}
 \tkzDefSpcTriangle[extouch,name=T](A,B,C){_a,_b,_c}
 \tkzDrawPolygon(A,B,C)
 \tkzDrawPolygon[new](J_a,J_b,J_c)
 \tkzClipBB
 \tkzDrawPoints(A,B,C)
 \tkzDrawPoints[new](J_a,J_b,J_c)
 \tkzLabelPoints(A,B,C)
 \tkzLabelPoints[new](J_b,J_c)
 \tkzLabelPoints[new,above](J_a)
 \tkzDrawCircles[gray](J_a,T_a J_b,T_b J_c,T_c)
\end{tikzpicture}
```

15.2.5. Option intouch or contact

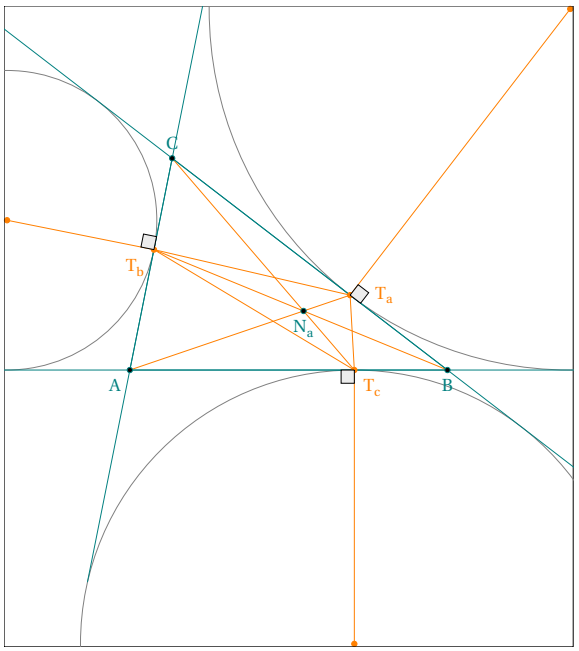
The contact triangle of a triangle ABC, also called the intouch triangle, is the triangle formed by the points of tangency of the incircle of ABC with its sides.
Weisstein, Eric W. "Contact triangle" From MathWorld–A Wolfram Web Resource.
We get the points where the incircle touches the sides.




```
\begin{tikzpicture}[scale=.75]
 \tkzDefPoints{0/0/A,6/0/B,0.8/4/C}
 \tkzDefSpcTriangle[intouch,name=X](A,B,C){_a,_b,_c}
 \tkzInCenter(A,B,C)\tkzGetPoint{I}
 \tkzDefCircle[in](A,B,C) \tkzGetPoints{I}{i}
 \tkzDrawCircle(I,i)
 \tkzDrawPolygon(A,B,C)
 \tkzDrawPolygon[new](X_a,X_b,X_c)
 \tkzDrawPoints(A,B,C)
 \tkzDrawPoints[new](X_a,X_b,X_c)
 \tkzLabelPoints[right](X_a)
 \tkzLabelPoints[left](X_b)
 \tkzLabelPoints[above](C)
 \tkzLabelPoints[below](A,B,X_c)
\end{tikzpicture}
```

15.2.6. Option extouch

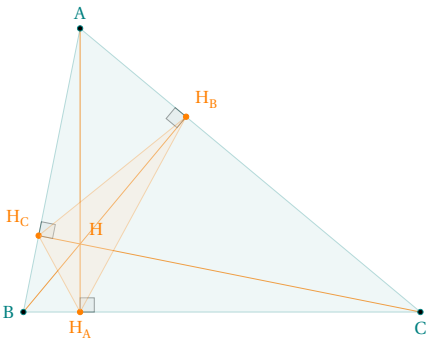
The extouch triangle $T_aT_bT_c$ is the triangle formed by the points of tangency of the sides of a triangle ABC with its excircles, with centers J_a, J_b and J_c. The points T_a, T_b and T_c can also be constructed as the points which bisect the perimeter of ABC starting at A, B and C.
Weisstein, Eric W. "Extouch triangle" From MathWorld–A Wolfram Web Resource.
We get the points of contact of the excircles, and the triangle formed by the centers of the excircles.




```
\begin{tikzpicture}[scale=.7]
\tkzDefPoints{0/0/A,6/0/B,0.8/4/C}
\tkzDefSpcTriangle[excentral,
                   name=J](A,B,C){_a,_b,_c}
\tkzDefSpcTriangle[extouch,
                    name=T](A,B,C){_a,_b,_c}
\tkzDefTriangleCenter[nagel](A,B,C)
\tkzGetPoint{N_a}
\tkzDefTriangleCenter[centroid](A,B,C)
\tkzGetPoint{G}
\tkzDrawPoints[new](J_a,J_b,J_c)
\tkzClipBB \tkzShowBB
\tkzDrawCircles[gray](J_a,T_a J_b,T_b J_c,T_c)
\tkzDrawLines[add=1 and 1](A,B B,C C,A)
\tkzDrawSegments[new](A,T_a B,T_b C,T_c)
\tkzDrawSegments[new](J_a,T_a J_b,T_b J_c,T_c)
\tkzDrawPolygon(A,B,C)
\tkzDrawPolygon[new](T_a,T_b,T_c)
\tkzDrawPoints(A,B,C,N_a)
\tkzDrawPoints[new](T_a,T_b,T_c)
\tkzLabelPoints[below left](A)
\tkzLabelPoints[below](N_a,B)
\tkzLabelPoints[above](C)
\tkzLabelPoints[new,below left](T_b)
\tkzLabelPoints[new,below right](T_c)
\tkzLabelPoints[new,right=6pt](T_a)
\tkzMarkRightAngles[fill=gray!15](J_a,T_a,B
 J_b,T_b,C J_c,T_c,A)
\end{tikzpicture}
```

15.2.7. Option orthic

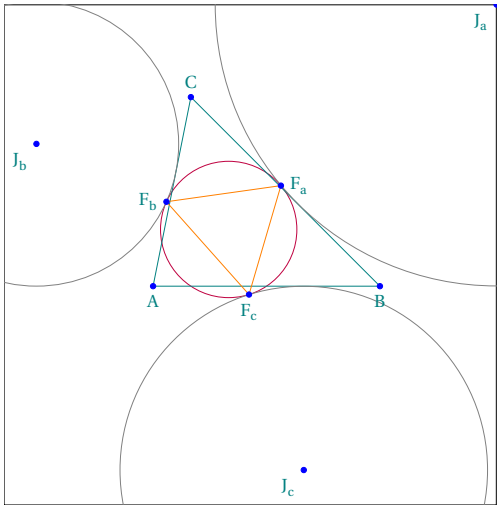
Given a triangle ABC, the triangle $H_AH_BH_C$ whose vertices are endpoints of the altitudes from each of the vertices of ABC is called the orthic triangle, or sometimes the altitude triangle. The three lines AH_A, BH_B and CH_C are concurrent at the orthocenter H of ABC.




```
\begin{tikzpicture}[scale=.75]
\tkzDefPoints{1/5/A,0/0/B,7/0/C}
 \tkzDefSpcTriangle[orthic](A,B,C){H_A,H_B,H_C}
 \tkzDefTriangleCenter[ortho](B,C,A)
 \tkzGetPoint{H}
 \tkzDefPointWith[orthogonal,normed](H_A,B)
 \tkzGetPoint{a}
 \tkzDrawSegments[new](A,H_A B,H_B C,H_C)
 \tkzMarkRightAngles[fill=gray!20,
        opacity=.5](A,H_A,C B,H_B,A C,H_C,A)
 \tkzDrawPolygon[fill=teal!20,opacity=.3](A,B,C)
 \tkzDrawPoints(A,B,C)
 \tkzDrawPoints[new](H_A,H_B,H_C)
 \tkzDrawPolygon[new,fill=orange!20,
                opacity=.3](H_A,H_B,H_C)
 \tkzLabelPoints(C)
 \tkzLabelPoints[left](B)
 \tkzLabelPoints[above](A)
 \tkzLabelPoints[new](H_A)
 \tkzLabelPoints[new,above left](H_C)
 \tkzLabelPoints[new,above right](H_B,H)
\end{tikzpicture}
```

15.2.8. Option feuerbach

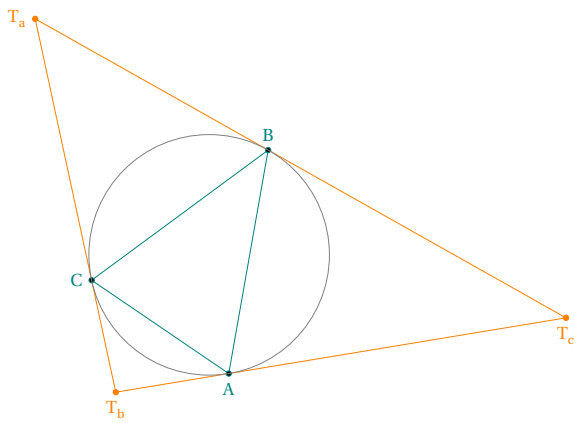
The Feuerbach triangle is the triangle formed by the three points of tangency of the nine-point circle with the excircles.
Weisstein, Eric W. "Feuerbach triangle" From MathWorld–A Wolfram Web Resource.
The points of tangency define the Feuerbach triangle.




```
\begin{tikzpicture}[scale=1]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(3,0){B}
  \tkzDefPoint(0.5,2.5){C}
  \tkzDefCircle[euler](A,B,C) \tkzGetPoint{N}
  \tkzDefSpcTriangle[feuerbach,
                     name=F](A,B,C){_a,_b,_c}
  \tkzDefSpcTriangle[excentral,
                     name=J](A,B,C){_a,_b,_c}
  \tkzDefSpcTriangle[extouch,
                      name=T](A,B,C){_a,_b,_c}
  \tkzLabelPoints[below left](J_a,J_b,J_c)
  \tkzClipBB \tkzShowBB
  \tkzDrawCircle[purple](N,F_a)
  \tkzDrawPolygon(A,B,C)
  \tkzDrawPolygon[new](F_a,F_b,F_c)
  \tkzDrawCircles[gray](J_a,F_a J_b,F_b J_c,F_c)
  \tkzDrawPoints[blue](J_a,J_b,J_c,%
         F_a,F_b,F_c,A,B,C)
  \tkzLabelPoints(A,B,F_c)
  \tkzLabelPoints[above](C)
  \tkzLabelPoints[right](F_a)
  \tkzLabelPoints[left](F_b)
\end{tikzpicture}
```

15.2.9. Option tangential

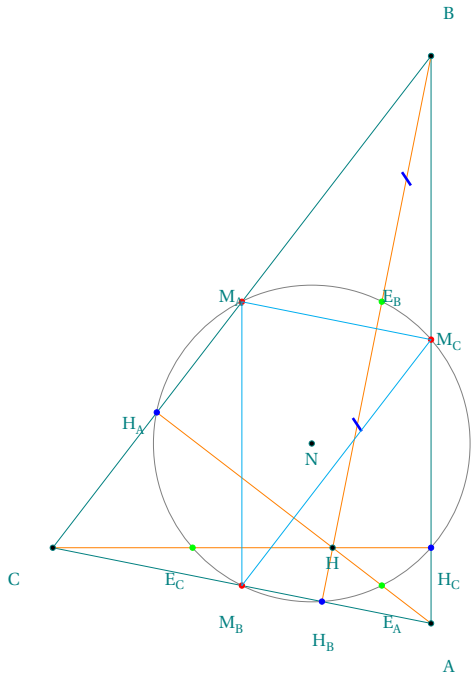
The tangential triangle is the triangle $T_aT_bT_c$ formed by the lines tangent to the circumcircle of a given triangle ABC at its vertices. It is therefore the antipedal triangle of ABC with respect to the circumcenter O.
Weisstein, Eric W. "Tangential Triangle." From MathWorld–A Wolfram Web Resource.




```
\begin{tikzpicture}[scale=.5,rotate=80]
  \tkzDefPoints{0/0/A,6/0/B,1.8/4/C}
  \tkzDefSpcTriangle[tangential,
    name=T](A,B,C){_a,_b,_c}
  \tkzDrawPolygon(A,B,C)
  \tkzDrawPolygon[new](T_a,T_b,T_c)
  \tkzDrawPoints(A,B,C)
  \tkzDrawPoints[new](T_a,T_b,T_c)
  \tkzDefCircle[circum](A,B,C)
  \tkzGetPoint{O}
  \tkzDrawCircle(O,A)
  \tkzLabelPoints(A)
  \tkzLabelPoints[above](B)
  \tkzLabelPoints[left](C)
  \tkzLabelPoints[new](T_b,T_c)
  \tkzLabelPoints[new,left](T_a)
\end{tikzpicture}
```

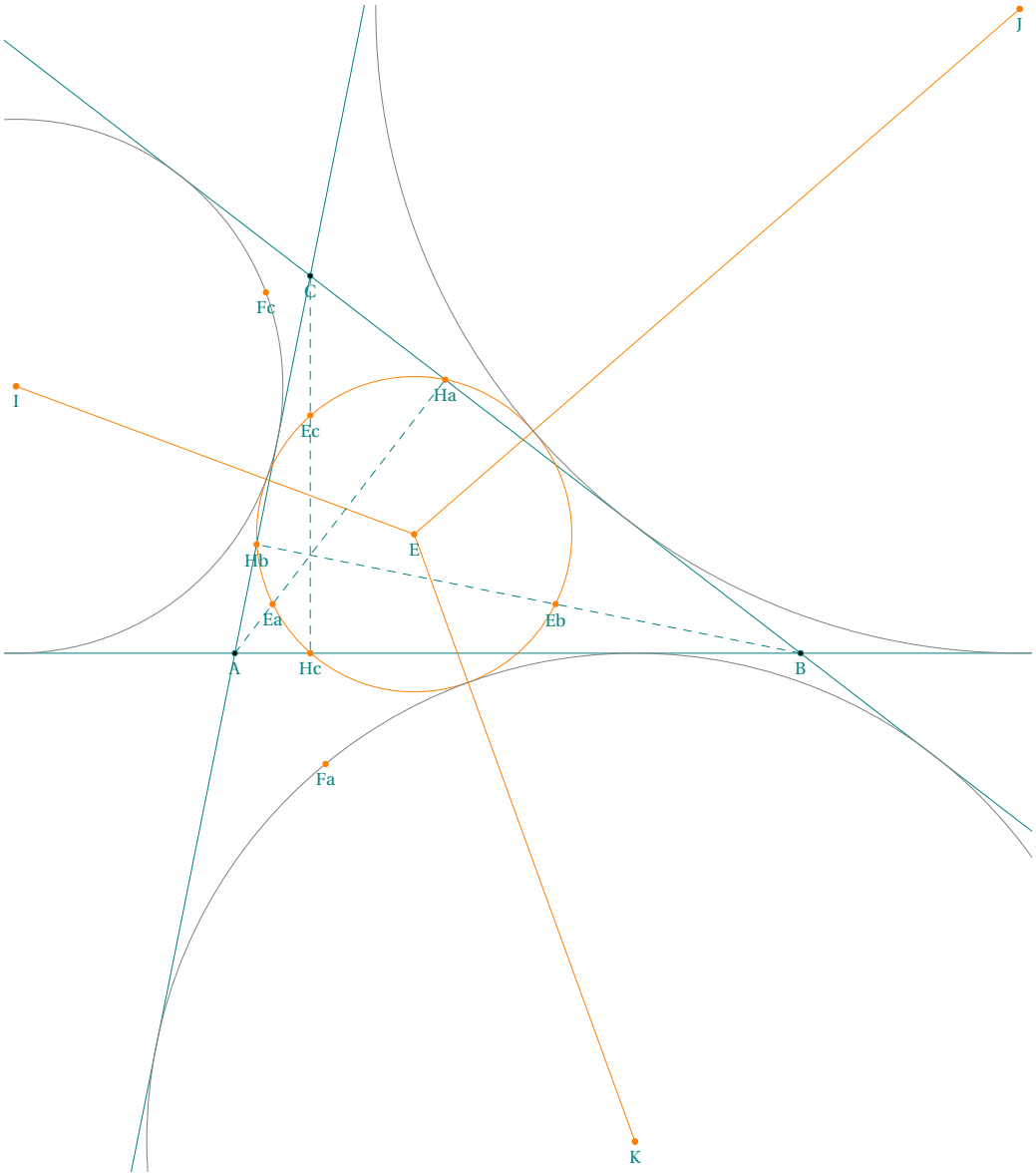
15.2.10. Option euler

The Euler triangle of a triangle ABC is the triangle $E_AE_BE_C$ whose vertices are the midpoints of the segments joining the orthocenter H with the respective vertices. The vertices of the triangle are known as the Euler points, and lie on the nine-point circle.
Weisstein, Eric W. "Euler Triangle." From MathWorld–A Wolfram Web Resource.




```
\begin{tikzpicture}[rotate=90,scale=1.25]
 \tkzDefPoints{0/0/A,6/0/B,0.8/4/C}
 \tkzDefSpcTriangle[medial,
     name=M](A,B,C){_A,_B,_C}
 \tkzDefTriangleCenter[euler](A,B,C)
     \tkzGetPoint{N} % I= N nine points
 \tkzDefTriangleCenter[ortho](A,B,C)
        \tkzGetPoint{H}
 \tkzDefMidPoint(A,H) \tkzGetPoint{E_A}
 \tkzDefMidPoint(C,H) \tkzGetPoint{E_C}
 \tkzDefMidPoint(B,H) \tkzGetPoint{E_B}
 \tkzDefSpcTriangle[ortho,name=H](A,B,C){_A,_B,_C}
 \tkzDrawPolygon(A,B,C)
 \tkzDrawCircle(N,E_A)
 \tkzDrawSegments[new](A,H_A B,H_B C,H_C)
 \tkzDrawPoints(A,B,C,N,H)
 \tkzDrawPoints[red](M_A,M_B,M_C)
 \tkzDrawPoints[blue]( H_A,H_B,H_C)
 \tkzDrawPoints[green](E_A,E_B,E_C)
 \tkzAutoLabelPoints[center=N,font=\scriptsize]%
(A,B,C,M_A,M_B,M_C,H_A,H_B,H_C,E_A,E_B,E_C)
\tkzLabelPoints[font=\scriptsize](H,N)
\tkzMarkSegments[mark=s|,size=3pt,
  color=blue,line width=1pt](B,E_B E_B,H)
   \tkzDrawPolygon[color=cyan](M_A,M_B,M_C)
\end{tikzpicture}
```

15.2.11. Option euler and Option orthic

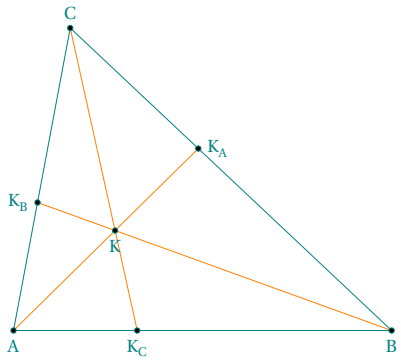



```
\begin{tikzpicture}[scale=1.25]
  \tkzDefPoints{0/0/A,6/0/B,0.8/4/C}
  \tkzDefSpcTriangle[euler,name=E](A,B,C){a,b,c}
  \tkzDefSpcTriangle[orthic,name=H](A,B,C){a,b,c}
  \tkzDefExCircle(A,B,C) \tkzGetPoints{I}{i}
  \tkzDefExCircle(C,A,B) \tkzGetPoints{J}{j}
  \tkzDefExCircle(B,C,A) \tkzGetPoints{K}{k}
  \tkzDrawPoints[orange](I,J,K)
  \tkzLabelPoints[font=\scriptsize](A,B,C,I,J,K)
  \tkzClipBB
  \tkzInterLC(I,C)(I,i) \tkzGetSecondPoint{Fc}
  \tkzInterLC(J,B)(J,j) \tkzGetSecondPoint{Fb}
  \tkzInterLC(K,A)(K,k) \tkzGetSecondPoint{Fa}
  \tkzDrawLines[add=1.5 and 1.5](A,B A,C B,C)
  \tkzDefCircle[euler](A,B,C) \tkzGetPoints{E}{e}
```

```
    \tkzDrawCircle[orange](E,e)
    \tkzDrawSegments[orange](E,I E,J E,K)
    \tkzDrawSegments[dashed](A,Ha B,Hb C,Hc)
    \tkzDrawCircles(J,j I,i K,k)
    \tkzDrawPoints(A,B,C)
    \tkzDrawPoints[orange](E,I,J,K,Ha,Hb,Hc,Ea,Eb,Ec,Fa,Fb,Fc)
    \tkzLabelPoints[font=\scriptsize](E,Ea,Eb,Ec,Ha,Hb,Hc,Fa,Fb,Fc)
  \end{tikzpicture}
```

15.2.12. Option symmedial

The symmedial triangle $K_AK_BK_C$ is the triangle whose vertices are the intersection points of the symmedians with the reference triangle ABC.




```
\begin{tikzpicture}
\tkzDefPoint(0,0){A}
\tkzDefPoint(5,0){B}
\tkzDefPoint(.75,4){C}
\tkzDefTriangleCenter[symmedian](A,B,C)\tkzGetPoint{K}
\tkzDefSpcTriangle[symmedial,name=K_](A,B,C){A,B,C}
\tkzDrawPolygon(A,B,C)
\tkzDrawSegments[new](A,K_A B,K_B C,K_C)
\tkzDrawPoints(A,B,C,K,K_A,K_B,K_C)
\tkzLabelPoints(A,B,K,K_C)
\tkzLabelPoints[above](C)
\tkzLabelPoints[right](K_A)
\tkzLabelPoints[left](K_B)
\end{tikzpicture}
```

15.3. Permutation of two points of a triangle

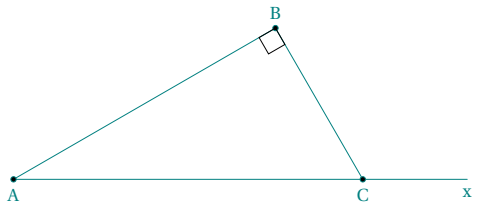
\tkzPermute(⟨pt1,pt2,pt3⟩)

arguments	example	explanation
(pt1,pt2,pt3)	\tkzPermute(A,B,C)	A and $\overset{\frown}{B,A,C}$ unchanged, B and C exchanged

The triangle is unchanged.

15.3.1. Modification of the school triangle

This triangle is constructed from the segment [AB] on the ray [Ax).
If we want the segment [AC] to be on [Ax), we just have to swap B and C.




```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,4/0/B,6/0/x}
  \tkzDefTriangle[school](A,B)
  \tkzGetPoint{C}
  \tkzPermute(A,B,C)
  \tkzDrawSegments(A,B C,x)
  \tkzDrawSegments(A,C B,C)
  \tkzDrawPoints(A,B,C)
  \tkzLabelPoints(A,C,x)
  \tkzLabelPoints[above](B)
  \tkzMarkRightAngles(C,B,A)
\end{tikzpicture}
```

Remark: only the first point is unchanged. The order of the last two arguments does not matter.

16. Definition of polygons

16.1. Defining the points of a square

We have seen how to define some triangles. Let us look at some quadrilaterals and regular polygons.

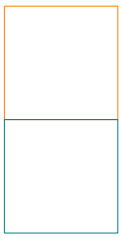
\tkzDefSquare(⟨pt1,pt2⟩)

From two points, two more points are obtained such that the four points, taken in order, form a square; the square is defined counterclockwise.
The results are in **tkzFirstPointResult** and **tkzSecondPointResult**.
We can rename them with **\tkzGetPoints**.

Arguments	example	explanation
(⟨pt1,pt2⟩)	**\tkzDefSquare**(⟨A,B⟩)	The square is defined in the direct direction.

16.1.1. Using \tkzDefSquare with two points

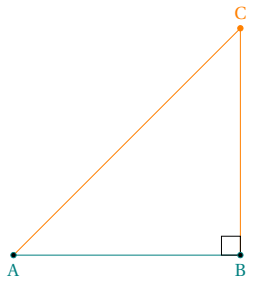
Note how exchanging the first two points changes the result.



```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoint(0,0){A} \tkzDefPoint(3,0){B}
  \tkzDefSquare(A,B)
  \tkzDrawPolygon[new](A,B,tkzFirstPointResult,%
                tkzSecondPointResult)
  \tkzDefSquare(B,A)
  \tkzDrawPolygon(B,A,tkzFirstPointResult,%
                tkzSecondPointResult)
\end{tikzpicture}
```

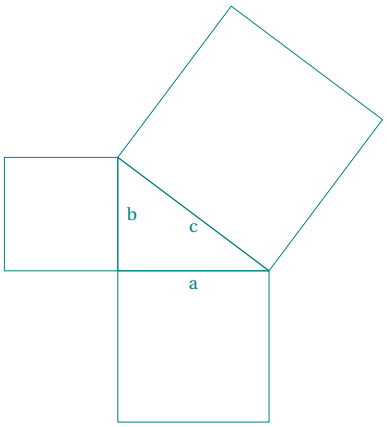
We may only need one point, to draw an isosceles right triangle, so we use **\tkzGetFirstPoint** or **\tkzGetSecondPoint**.

16.1.2. Use of \tkzDefSquare to obtain an isosceles right-angled triangle




```
\begin{tikzpicture}[scale=1]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(3,0){B}
  \tkzDefSquare(A,B) \tkzGetFirstPoint{C}
  \tkzDrawSegment(A,B)
  \tkzDrawSegments[new](A,C B,C)
  \tkzMarkRightAngles(A,B,C)
  \tkzDrawPoints(A,B) \tkzDrawPoint[new](C)
  \tkzLabelPoints(A,B)
  \tkzLabelPoints[new,above](C)
\end{tikzpicture}
```

16.1.3. Pythagorean Theorem and \tkzDefSquare




```
\begin{tikzpicture}[scale=.5]
\tkzDefPoint(0,0){C}
\tkzDefPoint(4,0){A}
\tkzDefPoint(0,3){B}
\tkzDefSquare(B,A)\tkzGetPoints{E}{F}
\tkzDefSquare(A,C)\tkzGetPoints{G}{H}
\tkzDefSquare(C,B)\tkzGetPoints{I}{J}
\tkzDrawPolygon(A,B,C)
\tkzDrawPolygon(A,C,G,H)
\tkzDrawPolygon(C,B,I,J)
\tkzDrawPolygon(B,A,E,F)
\tkzLabelSegment(A,C){$a$}
\tkzLabelSegment[right](C,B){$b$}
\tkzLabelSegment[swap](A,B){$c$}
\end{tikzpicture}
```

16.2. Defining the points of a rectangle

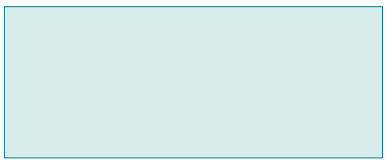
.

\tkzDefRectangle(⟨pt1,pt2⟩)

From two points, two more points are obtained such that the four points, taken in order, form a rectangle; the rectangle is defined counterclockwise. The two points given as arguments are the endpoints of a diagonal, and the sides are parallel to the axes.
The results are in **tkzFirstPointResult** and **tkzSecondPointResult**.
We can rename them with **\tkzGetPoints**.

Arguments	example	explanation
(⟨pt1,pt2⟩)	**\tkzDefRectangle**(⟨A,B⟩)	The rectangle is defined in the direct direction.

16.2.1. Example of a rectangle definition



```
\begin{tikzpicture}
\tkzDefPoints{0/0/A,5/2/C}
\tkzDefRectangle(A,C) \tkzGetPoints{B}{D}
\tkzDrawPolygon[fill=teal!15](A,...,D)
\end{tikzpicture}
```

16.3. Definition of parallelogram

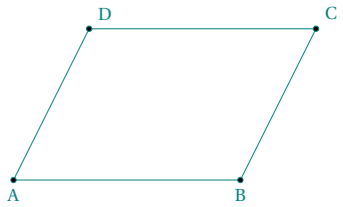
Defining the points of a parallelogram: a fourth point completes three given points.

\tkzDefParallelogram(⟨pt1,pt2,pt3⟩)

arguments	default	definition
(⟨pt1,pt2,pt3⟩)	no default	Three points are necessary

From three points, another point is obtained such that the four points, taken in order, form a parallelogram.
The result is in **tkzPointResult**.
We can rename it with **\tkzGetPoint**.

16.3.1. Example of a parallelogram definition




```
\begin{tikzpicture}[scale=1]
 \tkzDefPoints{0/0/A,3/0/B,4/2/C}
 \tkzDefParallelogram(A,B,C)
 % or   \tkzDefPointWith[colinear= at C](B,A)
 \tkzGetPoint{D}
 \tkzDrawPolygon(A,B,C,D)
 \tkzLabelPoints(A,B)
 \tkzLabelPoints[above right](C,D)
 \tkzDrawPoints(A,...,D)
\end{tikzpicture}
```

16.4. The golden rectangle

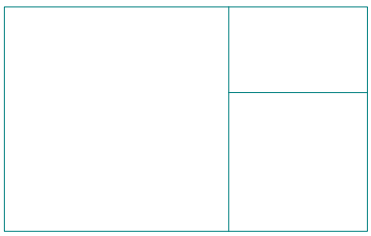
\tkzDefGoldenRectangle(⟨point,point⟩)

The macro determines a rectangle whose sides are in the ratio Φ.
The created points are in **tkzFirstPointResult** and **tkzSecondPointResult**.
They can be retrieved with the macro **\tkzGetPoints**.

arguments	example	explanation
(⟨pt1,pt2⟩)	(⟨A,B⟩)	If C and D are created then AB/BC = Φ.

\tkzDefGoldenRectangle or **\tkzDefGoldRectangle**

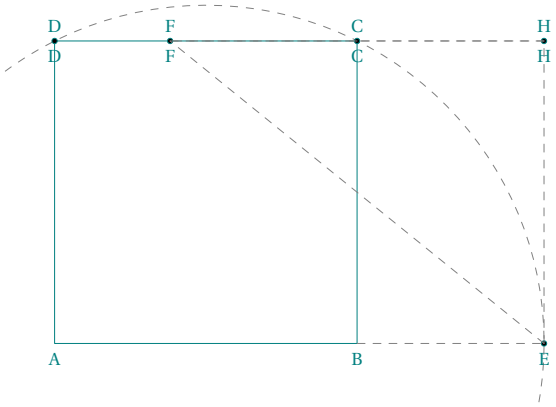
16.4.1. Golden Rectangles



```
\begin{tikzpicture}[scale=.6]
 \tkzDefPoint(0,0){A}      \tkzDefPoint(8,0){B}
 \tkzDefGoldRectangle(A,B) \tkzGetPoints{C}{D}
 \tkzDefGoldRectangle(B,C) \tkzGetPoints{E}{F}
 \tkzDefGoldRectangle(C,E) \tkzGetPoints{G}{H}
 \tkzDrawPolygon(A,B,C,D)
 \tkzDrawSegments(E,F G,H)
\end{tikzpicture}
```

16.4.2. Construction of the golden rectangle

Here is how to get the golden rectangle without the previous macro.




```
\begin{tikzpicture}[scale=.5]
\tkzDefPoint(0,0){A}
\tkzDefPoint(8,0){B}
\tkzDefMidPoint(A,B)
\tkzGetPoint{I}
\tkzDefSquare(A,B)\tkzGetPoints{C}{D}
\tkzInterLC(A,B)(I,C)\tkzGetPoints{G}{E}
\tkzDefPointWith[colinear= at C](E,B)
 \tkzGetPoint{F}
\tkzDefPointBy[projection=onto D--C ](E)
 \tkzGetPoint{H}
\tkzDrawArc[style=dashed](I,E)(D)
\tkzDrawPolygon(A,B,C,D)
\tkzDrawPoints(C,D,E,F,H)
\tkzLabelPoints(A,B,C,D,E,F,H)
\tkzLabelPoints[above](C,D,F,H)
\tkzDrawSegments[style=dashed,color=gray]%
(E,F C,F B,E F,H H,C E,H)
\end{tikzpicture}
```

16.5. Regular polygon

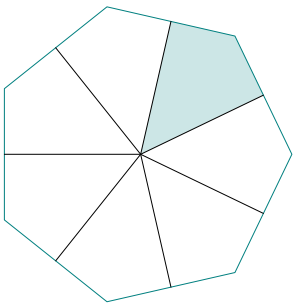
\tkzDefRegPolygon[⟨local options⟩](⟨pt1,pt2⟩)

Given the number of sides, this macro determines a regular polygon from its center or from one side, depending on the option.

arguments	example	explanation
(⟨pt1,pt2⟩)	(⟨O,A⟩)	with option center, O is the center of the polygon.
(⟨pt1,pt2⟩)	(⟨A,B⟩)	with option side, [AB] is a side.

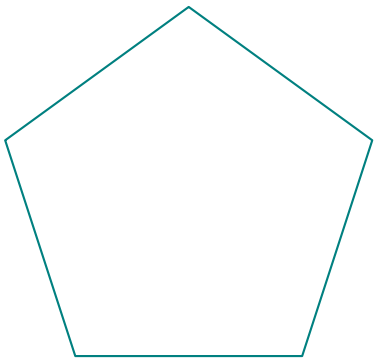
options	default	example
name	P	The vertices are named P1,P2,…
sides	5	number of sides.
center	center	The first point is the center.
side	center	The two points are vertices.
Options TikZ	...	

16.5.1. Option center



```
\begin{tikzpicture}
  \tkzDefPoints{0/0/P0,0/0/Q0,2/0/P1}
  \tkzDefMidPoint(P0,P1) \tkzGetPoint{Q1}
  \tkzDefRegPolygon[center,sides=7](P0,P1)
  \tkzDefMidPoint(P1,P2) \tkzGetPoint{Q1}
  \tkzDefRegPolygon[center,sides=7,name=Q](P0,Q1)
  \tkzFillPolygon[teal!20](Q0,Q1,P2,Q2)
  \tkzDrawPolygon(P1,P...,P7)
  \foreach \j in {1,...,7} {%
  \tkzDrawSegment[black](P0,Q\j)}
\end{tikzpicture}
```

16.5.2. Option side



```
\begin{tikzpicture}[scale=1]
    \tkzDefPoints{-4/0/A, -1/0/B}
    \tkzDefRegPolygon[side,sides=5,name=P](A,B)
    \tkzDrawPolygon[thick](P1,P...,P5)
\end{tikzpicture}
```

Part IV.

Circles

17. Circles

Among the following macros, one draws a circle, which is not a real feat: you need the center of the circle and either its radius or a point of the circle. The most frequent use seemed to me to be a circle with a given center through a given point; this is the default, otherwise you use the option R. There are many special circles, for example the circumcircle of a triangle.

- a first macro, \tkzDefCircle, gives the center and the radius (in cm) of a particular circle, which you retrieve with the macros \tkzGetPoint and \tkzGetLength;
- a macro to draw a circle, \tkzDrawCircle;
- a macro to color a disc without drawing the circle, \tkzFillCircle;
- sometimes a drawing must be contained in a disc: this is the role of \tkzClipCircle;
- finally, \tkzLabelCircle gives a label to a circle.

17.1. Characteristics of a circle: \tkzDefCircle

This macro gives the characteristics (center and radius) of some circles.

\tkzDefCircle[⟨local options⟩](⟨A,B⟩) or (⟨A,B,C⟩)

Note that the arguments are lists of two or three points. The macro is used either with \tkzGetPoints, to get the center and a point of the circle, or directly with tkzFirstPointResult and tkzSecondPointResult if you don't need to keep the results. \tkzGetLength gives the radius.

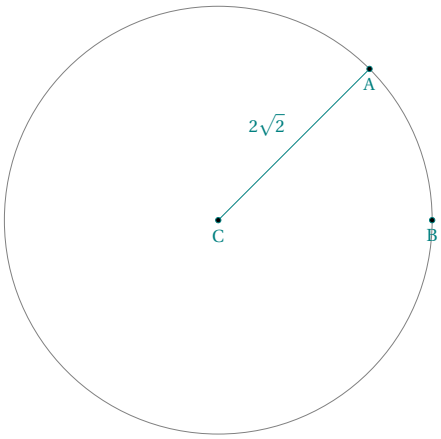
arguments	example	explanation
(⟨pt1,pt2⟩) or (⟨pt1,pt2,pt3⟩)	(⟨A,B⟩)	[AB] is radius A is the center

options	default	definition
R	R	circle characterized by a center and a radius
diameter		circle characterized by two points defining a diameter
circum		circle circumscribed of a triangle
in		incircle a triangle
ex		excircle of a triangle
euler or nine		Euler's Circle
spieker		Spieker Circle
apollonius		circle of Apollonius
orthogonal from		[orthogonal from = A](O,M)
orthogonal through		[orthogonal through = A and B](O,M)
K	1	coefficient used for a circle of Apollonius

In the following examples, I draw the circles with a macro not yet presented. You may only need the center and a point of the circle.

17.1.1. Example with option R

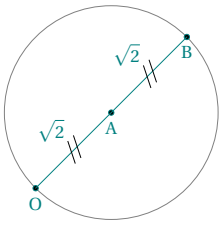
With the macro \tkzGetPoint we get a point of the circle, its "east pole".




```
\begin{tikzpicture}[scale=1]
  \tkzDefPoint(3,3){C}
  \tkzDefPoint(5,5){A}
   \tkzCalcLength(A,C) \tkzGetLength{rAC}
  \tkzDefCircle[R](C,\rAC) \tkzGetPoint{B}
  \tkzDrawCircle(C,B)
  \tkzDrawSegment(C,A)
  \tkzLabelSegment[above left](C,A){$2\sqrt{2}$}
  \tkzDrawPoints(A,B,C)
  \tkzLabelPoints(A,C,B)
\end{tikzpicture}
```

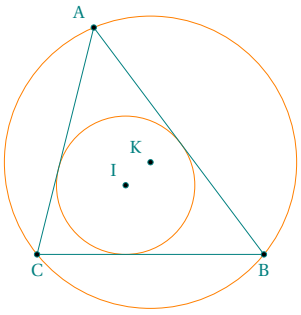
17.1.2. Example with option diameter

It is simpler here to search directly for the midpoint of [AB]. The result is the center and if necessary




```
\begin{tikzpicture}
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2,2){B}
  \tkzDefCircle[diameter](O,B) \tkzGetPoint{A}
  \tkzDrawCircle(A,B)
  \tkzDrawPoints(O,A,B)
  \tkzDrawSegment(O,B)
  \tkzLabelPoints(O,A,B)
  \tkzLabelSegment[above left](O,A){$\sqrt{2}$}
  \tkzLabelSegment[above left](A,B){$\sqrt{2}$}
  \tkzMarkSegments[mark=s||](O,A A,B)
\end{tikzpicture}
```

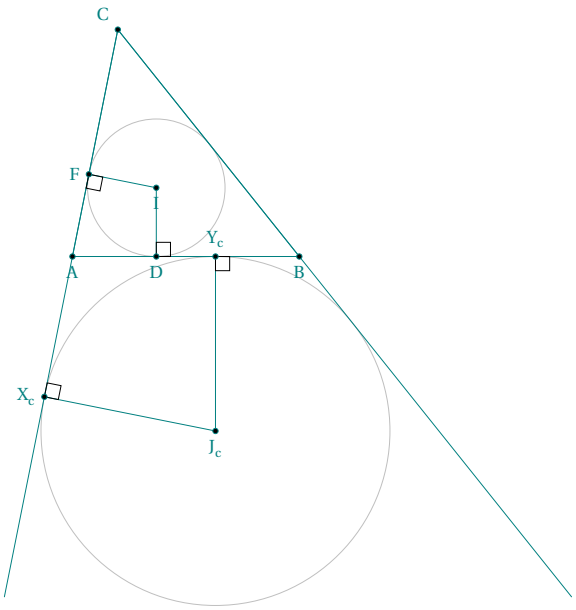
17.1.3. Circles inscribed and circumscribed for a given triangle




```
\begin{tikzpicture}[scale=.75]
 \tkzDefPoint(2,2){A}  \tkzDefPoint(5,-2){B}
 \tkzDefPoint(1,-2){C}
 \tkzDefCircle[in](A,B,C)
 \tkzGetPoints{I}{x}
 \tkzDefCircle[circum](A,B,C)
 \tkzGetPoint{K}
 \tkzDrawCircles[new](I,x K,A)
 \tkzLabelPoints[below](B,C)
 \tkzLabelPoints[above left](A,I,K)
 \tkzDrawPolygon(A,B,C)
 \tkzDrawPoints(A,B,C,I,K)
\end{tikzpicture}
```

17.1.4. Example with option ex

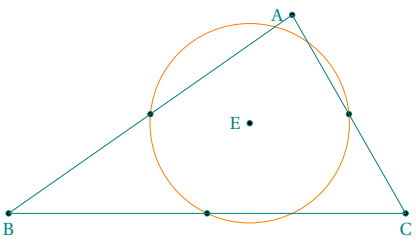
We want to define the excircle of a triangle opposite the vertex C.




```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoints{ 0/0/A,4/0/B,0.8/4/C}
  \tkzDefCircle[ex](B,C,A)
  \tkzGetPoints{J_c}{h}
  \tkzDefPointBy[projection=onto A--C ](J_c)
  \tkzGetPoint{X_c}
  \tkzDefPointBy[projection=onto A--B ](J_c)
  \tkzGetPoint{Y_c}
  \tkzDefCircle[in](A,B,C)
  \tkzGetPoints{I}{y}
  \tkzDrawCircles[color=lightgray](J_c,h I,y)
  \tkzDefPointBy[projection=onto A--C ](I)
  \tkzGetPoint{F}
  \tkzDefPointBy[projection=onto A--B ](I)
  \tkzGetPoint{D}
  \tkzDrawPolygon(A,B,C)
  \tkzDrawLines[add=0 and 1.5](C,A C,B)
  \tkzDrawSegments(J_c,X_c I,D  I,F J_c,Y_c)
  \tkzMarkRightAngles(A,F,I B,D,I J_c,X_c,A
     J_c,Y_c,B)
  \tkzDrawPoints(B,C,A,I,D,F,X_c,J_c,Y_c)
  \tkzLabelPoints(B,A,J_c,I,D)
  \tkzLabelPoints[above](Y_c)
  \tkzLabelPoints[left](X_c)
  \tkzLabelPoints[above left](C)
  \tkzLabelPoints[left](F)
\end{tikzpicture}
```

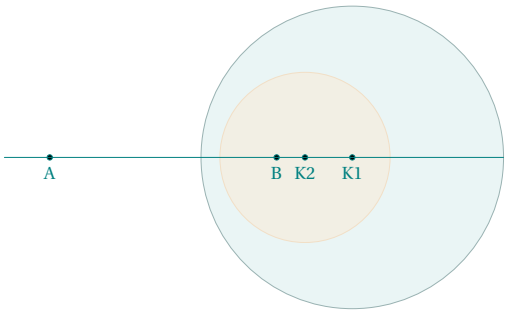
17.1.5. Euler's circle for a given triangle with option euler

We verify that this circle passes through the midpoint of each side.




```
\begin{tikzpicture}[scale=.75]
   \tkzDefPoint(5,3.5){A}
   \tkzDefPoint(0,0){B} \tkzDefPoint(7,0){C}
   \tkzDefCircle[euler](A,B,C)
   \tkzGetPoints{E}{e}
   \tkzDefSpcTriangle[medial](A,B,C){M_a,M_b,M_c}
   \tkzDrawCircle[new](E,e)
   \tkzDrawPoints(A,B,C,E,M_a,M_b,M_c)
   \tkzDrawPolygon(A,B,C)
   \tkzLabelPoints[below](B,C)
   \tkzLabelPoints[left](A,E)
\end{tikzpicture}
```

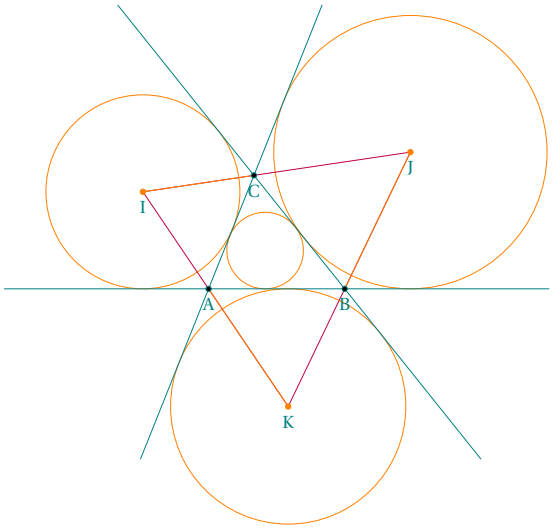
17.1.6. Apollonius circles for a given segment: option apollonius




```
\begin{tikzpicture}[scale=0.75]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(4,0){B}
  \tkzDefCircle[apollonius,K=2](A,B)
  \tkzGetPoints{K1}{x}
  \tkzDrawCircle[color = teal!50!black,
      fill=teal!20,opacity=.4](K1,x)
  \tkzDefCircle[apollonius,K=3](A,B)
  \tkzGetPoints{K2}{y}
  \tkzDrawCircle[color=orange!50,
      fill=orange!20,opacity=.4](K2,y)
  \tkzLabelPoints[below](A,B,K1,K2)
  \tkzDrawPoints(A,B,K1,K2)
  \tkzDrawLine[add=.2 and 1](A,B)
\end{tikzpicture}
```

17.1.7. Excircles of a given triangle: option ex

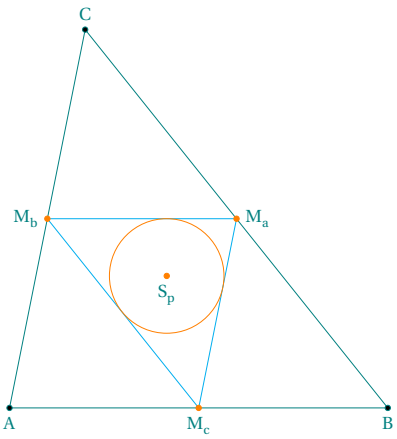
You can also get the center and its projection on one side of the triangle with `\tkzGetFirstPoint{Jb}` and `\tkzGetSecondPoint{Tb}`.




```
\begin{tikzpicture}[scale=.6]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(3,0){B}
  \tkzDefPoint(1,2.5){C}
  \tkzDefCircle[ex](A,B,C) \tkzGetPoints{I}{i}
  \tkzDefCircle[ex](C,A,B) \tkzGetPoints{J}{j}
  \tkzDefCircle[ex](B,C,A) \tkzGetPoints{K}{k}
  \tkzDefCircle[in](B,C,A) \tkzGetPoints{O}{o}
  \tkzDrawCircles[new](J,j I,i K,k O,o)
  \tkzDrawLines[add=1.5 and 1.5](A,B A,C B,C)
  \tkzDrawPolygon[purple](I,J,K)
  \tkzDrawSegments[new](A,K B,J C,I)
  \tkzDrawPoints(A,B,C)
  \tkzDrawPoints[new](I,J,K)
  \tkzLabelPoints(A,B,C,I,J,K)
\end{tikzpicture}
```

17.1.8. Spieker circle with option spieker

The incircle of the medial triangle $M_aM_bM_c$ is the Spieker circle:




```
\begin{tikzpicture}[scale=1.25]
  \tkzDefPoints{ 0/0/A,4/0/B,0.8/4/C}
  \tkzDefSpcTriangle[medial](A,B,C){M_a,M_b,M_c}
  \tkzDefTriangleCenter[spieker](A,B,C)
  \tkzGetPoint{S_p}
  \tkzDrawPolygon(A,B,C)
  \tkzDrawPolygon[cyan](M_a,M_b,M_c)
  \tkzDrawPoints(B,C,A)
  \tkzDefCircle[spieker](A,B,C)
  \tkzDrawPoints[new](M_a,M_b,M_c,S_p)
  \tkzDrawCircle[new](tkzFirstPointResult,%
    tkzSecondPointResult)
  \tkzLabelPoints[right](M_a)
  \tkzLabelPoints[left](M_b)
  \tkzLabelPoints[below](A,B,M_c,S_p)
  \tkzLabelPoints[above](C)
\end{tikzpicture}
```

17.2. Projection of excenters

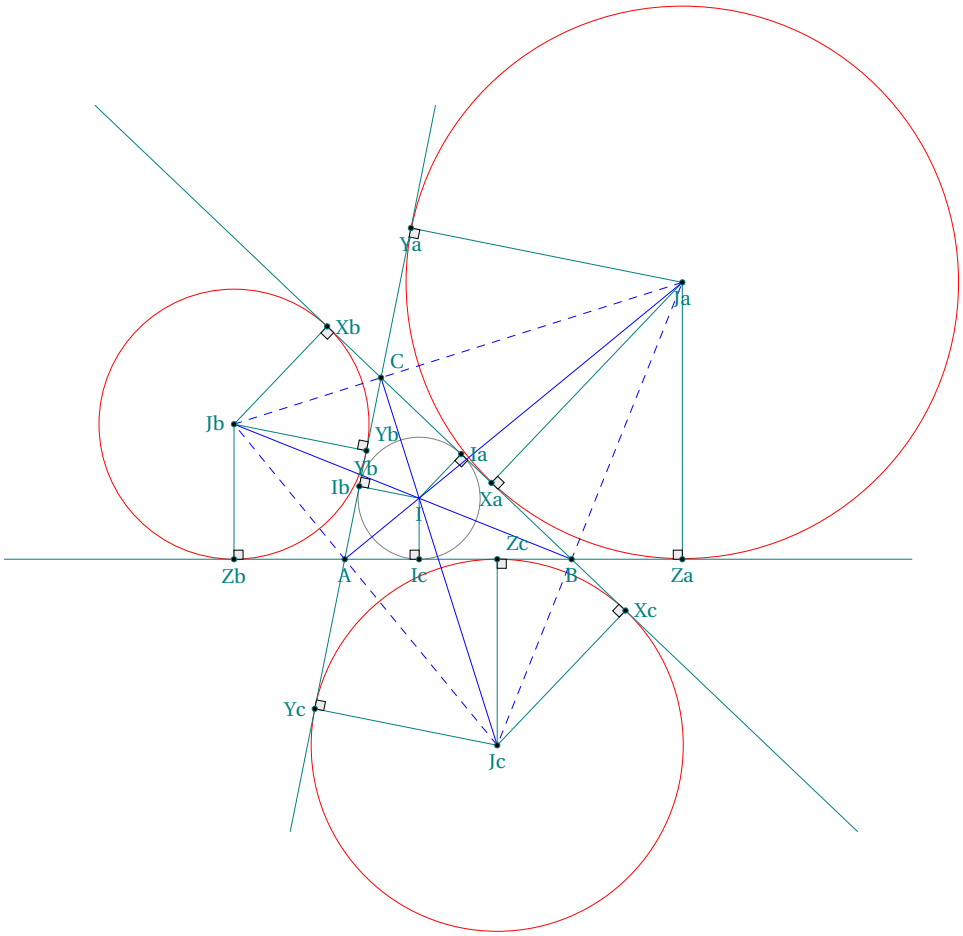
\tkzDefProjExcenter[⟨local options⟩](⟨A,B,C⟩)(⟨a,b,c⟩){⟨X,Y,Z⟩}

Each excenter has three projections on the sides of the triangle ABC. We can get them all with one macro: **\tkzDefProjExcenter[name=J](A,B,C)(a,b,c){Y,Z,X}**.

options	default	definition
name	no defaut	used to name the vertices

arguments	default	definition
(pt1=α_1,pt2=α_2,…)	no default	Each point has a assigned weight

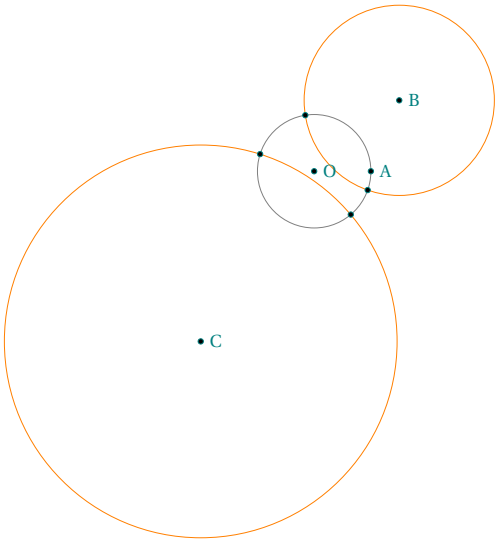
17.2.1. Excircles



```
\begin{tikzpicture}[scale=.6]
\tikzset{line style/.append style={line width=.2pt}}
\tikzset{label style/.append style={color=teal,font=\footnotesize}}
\tkzDefPoints{0/0/A,5/0/B,0.8/4/C}
\tkzDefSpcTriangle[excentral,name=J](A,B,C){a,b,c}
\tkzDefSpcTriangle[intouch,name=I](A,B,C){a,b,c}
\tkzDefProjExcenter[name=J](A,B,C)(a,b,c){X,Y,Z}
\tkzDefCircle[in](A,B,C)   \tkzGetPoint{I} \tkzGetSecondPoint{T}
\tkzDrawCircles[red](Ja,Xa Jb,Yb Jc,Zc)
\tkzDrawCircle(I,T)
\tkzDrawPolygon[dashed,color=blue](Ja,Jb,Jc)
\tkzDrawLines[add=1.5 and 1.5](A,C A,B B,C)
\tkzDrawSegments(Ja,Xa Ja,Ya Ja,Za
                Jb,Xb Jb,Yb Jb,Zb
                Jc,Xc Jc,Yc Jc,Zc
                I,Ia I,Ib I,Ic)
\tkzMarkRightAngles[size=.2,fill=gray!15](Ja,Za,B Ja,Xa,B Ja,Ya,C Jb,Yb,C)
\tkzMarkRightAngles[size=.2,fill=gray!15](Jb,Zb,B Jb,Xb,C Jc,Yc,A Jc,Zc,B)
\tkzMarkRightAngles[size=.2,fill=gray!15](Jc,Xc,C I,Ia,B I,Ib,C I,Ic,A)
\tkzDrawSegments[blue](Jc,C Ja,A Jb,B)
\tkzDrawPoints(A,B,C,Xa,Xb,Xc,Ja,Jb,Jc,Ia,Ib,Ic,Ya,Yb,Yc,Za,Zb,Zc)
\tkzLabelPoints(A,Ya,Yb,Ja,I)
\tkzLabelPoints[left](Jb,Ib,Yc)
\tkzLabelPoints[below](Zb,Ic,Jc,B,Za,Xa)
\tkzLabelPoints[above right](C,Zc,Yb)
\tkzLabelPoints[right](Xb,Ia,Xc)
\end{tikzpicture}
```

17.2.2. Orthogonal from

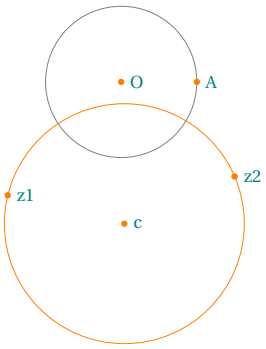
Circle with a given center, orthogonal to a given circle. **\tkzGetPoints{z1}{z2}** gives two points of the circle.




```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoints{0/0/O,1/0/A}
  \tkzDefPoints{1.5/1.25/B,-2/-3/C}
  \tkzDefCircle[orthogonal from=B](O,A)
  \tkzGetPoints{z1}{z2}
  \tkzDefCircle[orthogonal from=C](O,A)
  \tkzGetPoints{t1}{t2}
  \tkzDrawCircle(O,A)
  \tkzDrawCircles[new](B,z1 C,t1)
  \tkzDrawPoints(t1,t2,C)
  \tkzDrawPoints(z1,z2,O,A,B)
  \tkzLabelPoints[right](O,A,B,C)
\end{tikzpicture}
```

17.2.3. Orthogonal through

Circle orthogonal to a given circle, through two given points.




```
\begin{tikzpicture}[scale=1]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(1,0){A}
  \tkzDrawCircle(O,A)
  \tkzDefPoint(-1.5,-1.5){z1}
  \tkzDefPoint(1.5,-1.25){z2}
  \tkzDefCircle[orthogonal through=z1 and z2](O,A)
   \tkzGetPoint{c}
  \tkzDrawCircle[new](tkzPointResult,z1)
  \tkzDrawPoints[new](O,A,z1,z2,c)
  \tkzLabelPoints[right](O,A,z1,z2,c)
\end{tikzpicture}
```

18. Circles by transformation

18.1. The macro \tkzDefCircleBy

These transformations are:

- translation;
- homothety;
- orthogonal reflection or symmetry;
- central symmetry;
- orthogonal projection;
- rotation (degrees);
- inversion.

The transformation is chosen with the options. For example, we write:

```
\tkzDefCircleBy[translation= from A to A'](O,M)
```

O is the center and M a point of the circle. The image is a circle: the new center is `tkzFirstPointResult`, and `tkzSecondPointResult` is a point of the new circle. You get the results with the macro `\tkzGetPoints`. With an inversion, the image of a circle through the center of inversion is a line: the two points are then two points of this line.

`\tkzDefCircleBy[`⟨local options⟩`](`⟨pt1,pt2⟩`)`

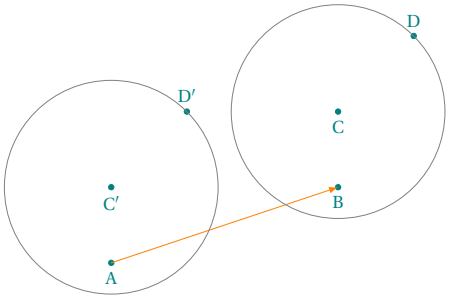
The argument is a pair of points, and so is the result. To keep these points, `\tkzGetPoints{O'}{M'}` gives the name `O'` to the center and `M'` to the point of the circle.

arguments	definition	examples
pt1,pt2	existing points	(O,M)

options		examples
translation	= from #1 to #2	[translation=from A to B](O,M)
homothety	= center #1 ratio #2	[homothety=center A ratio .5](O,M)
reflection	= over #1--#2	[reflection=over A--B](O,M)
symmetry	= center #1	[symmetry=center A](O,M)
projection	= onto #1--#2	[projection=onto A--B](O,M)
rotation	= center #1 angle #2	[rotation=center O angle 30](O,M)
inversion	= center #1 through #2	[inversion =center O through A](O,M)

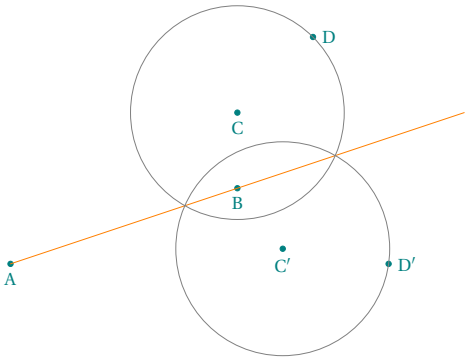
The image is only defined, not drawn.

18.1.1. Translation



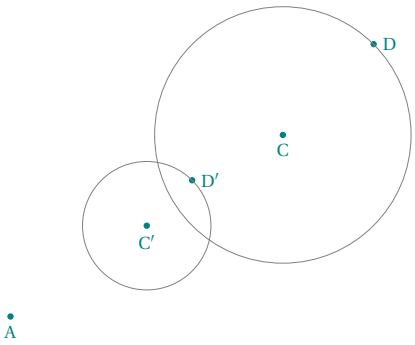

```
\begin{tikzpicture}[>=latex]
 \tkzDefPoint(0,0){A}  \tkzDefPoint(3,1){B}
 \tkzDefPoint(3,2){C}  \tkzDefPoint(4,3){D}
 \tkzDefCircleBy[translation= from B to A](C,D)
 \tkzGetPoints{C'}{D'}
 \tkzDrawPoints[teal](A,B,C,D,C',D')
 \tkzDrawSegments[orange,->](A,B)
 \tkzDrawCircles(C,D C',D')
 \tkzLabelPoints[color=teal](A,B,C,C')
 \tkzLabelPoints[color=teal,above](D,D')
\end{tikzpicture}
```

18.1.2. Reflection (orthogonal symmetry)



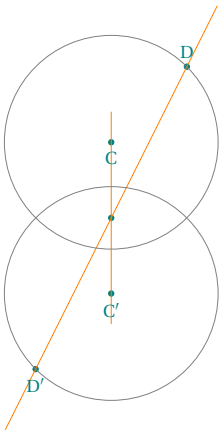

```
\begin{tikzpicture}[>=latex]
 \tkzDefPoint(0,0){A}  \tkzDefPoint(3,1){B}
 \tkzDefPoint(3,2){C}  \tkzDefPoint(4,3){D}
 \tkzDefCircleBy[reflection = over A--B](C,D)
 \tkzGetPoints{C'}{D'}
 \tkzDrawPoints[teal](A,B,C,D,C',D')
 \tkzDrawLine[add =0 and 1][orange](A,B)
 \tkzDrawCircles(C,D C',D')
 \tkzLabelPoints[color=teal](A,B,C,C')
 \tkzLabelPoints[color=teal,right](D,D')
\end{tikzpicture}
```

18.1.3. Homothety



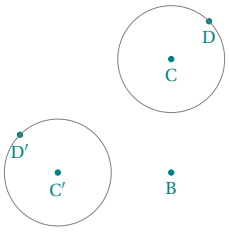

```
\begin{tikzpicture}[scale=1.2]
 \tkzDefPoint(0,0){A}  \tkzDefPoint(3,1){B}
 \tkzDefPoint(3,2){C}  \tkzDefPoint(4,3){D}
 \tkzDefCircleBy[homothety=center A ratio .5](C,D)
 \tkzGetPoints{C'}{D'}
 \tkzDrawPoints[teal](A,C,D,C',D')
 \tkzDrawCircles(C,D C',D')
 \tkzLabelPoints[color=teal](A,C,C')
 \tkzLabelPoints[color=teal,right](D,D')
\end{tikzpicture}
```

18.1.4. Symmetry



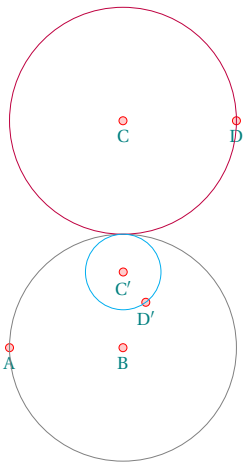

```
\begin{tikzpicture}[scale=1]
 \tkzDefPoint(3,1){B}
 \tkzDefPoint(3,2){C}   \tkzDefPoint(4,3){D}
 \tkzDefCircleBy[symmetry=center B](C,D)
 \tkzGetPoints{C'}{D'}
 \tkzDrawPoints[teal](B,C,D,C',D')
 \tkzDrawLines[orange](C,C' D,D')
 \tkzDrawCircles(C,D C',D')
 \tkzLabelPoints[color=teal](C,C')
 \tkzLabelPoints[color=teal,above](D)
 \tkzLabelPoints[color=teal,below](D')
\end{tikzpicture}
```

18.1.5. Rotation




```
\begin{tikzpicture}[scale=0.5]
 \tkzDefPoint(3,-1){B}
 \tkzDefPoint(3,2){C}   \tkzDefPoint(4,3){D}
 \tkzDefCircleBy[rotation=center B angle 90](C,D)
 \tkzGetPoints{C'}{D'}
 \tkzDrawPoints[teal](B,C,D,C',D')
 \tkzLabelPoints[color=teal](B,C,D,C',D')
 \tkzDrawCircles(C,D C',D')
\end{tikzpicture}
```

18.1.6. Inversion




```
\begin{tikzpicture}[scale=1.5]
\tkzSetUpPoint[size=3,color=red,fill=red!20]
\tkzSetUpStyle[color=purple,ultra thin]{st1}
\tkzSetUpStyle[color=cyan,ultra thin]{st2}
\tkzDefPoint(2,0){A} \tkzDefPoint(3,0){B}
\tkzDefPoint(3,2){C} \tkzDefPoint(4,2){D}
\tkzDefCircleBy[inversion = center B through A](C,D)
\tkzGetPoints{C'}{D'}
\tkzDrawPoints(A,B,C,D,C',D')
\tkzLabelPoints(A,B,C,D,C',D')
\tkzDrawCircles(B,A)
\tkzDrawCircles[st1](C,D)
\tkzDrawCircles[st2](C',D')
\end{tikzpicture}
```

Part V.

Intersections and angles

19. Intersections

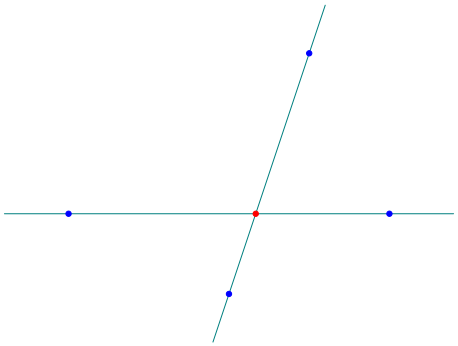
You can determine the points of intersection of two straight lines, of a straight line and a circle, and of two circles. The associated commands do not check that the intersection points exist: this is up to the user (see the tests **\tkzTestInterLC** and **\tkzTestInterCC**).

19.1. Intersection of two straight lines \tkzInterLL

\tkzInterLL(⟨A,B⟩)(⟨C,D⟩)

Defines the intersection point **tkzPointResult** of the two lines (AB) and (CD). The known points are given in pairs (two per line) between parentheses, and the resulting point is retrieved with the macro **\tkzGetPoint**.

19.1.1. Example of intersection between two straight lines



```
\begin{tikzpicture}[rotate=-45,scale=.75]
  \tkzDefPoint(2,1){A}
  \tkzDefPoint(6,5){B}
  \tkzDefPoint(3,6){C}
  \tkzDefPoint(5,2){D}
  \tkzDrawLines(A,B C,D)
  \tkzInterLL(A,B)(C,D)
     \tkzGetPoint{I}
  \tkzDrawPoints[color=blue](A,B,C,D)
   \tkzDrawPoint[color=red](I)
\end{tikzpicture}
```

19.2. Intersection of a straight line and a circle \tkzInterLC

As before, the line is defined by a pair of points. The circle is also defined by a pair:

- (O,C), a pair of points: the first one is the center and the second one is any point of the circle;
- (O,r), where r is the radius.

\tkzInterLC[⟨options⟩](⟨A,B⟩)(⟨O,C⟩) or (⟨O,r⟩) or (⟨O,C,D⟩)

So the arguments are two pairs.

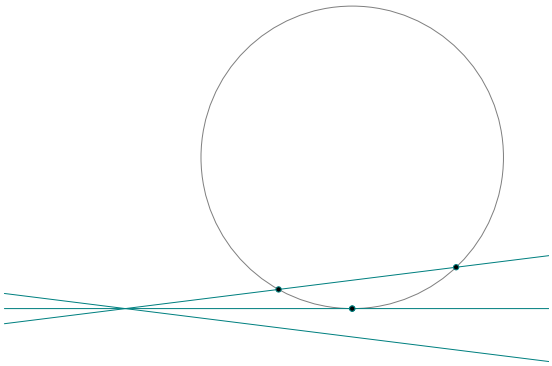
options	default	definition
N	N	(O,C) determines the circle
R	N	(O, 1) unit 1 cm
with nodes	N	(O,C,D) CD is a radius
common=pt		pt is common point; tkzFirstPoint gives the other point
near		tkzFirstPoint is the closest point to the first point of the line

The macro defines the intersection points I and J of the line (AB) and the circle with center O and radius r, if they exist; otherwise, an error is reported in the .log file. With **with nodes**, you don't need to calculate the radius, which is the length of [CD]. If **common** and **near** are not used, **tkzFirstPoint** is the point that gives the smallest angle (formed with **tkzSecondPoint** and the center of the circle). If the line passes through the center, **tkzSecondPoint** is the point located in the direction of the line, from A to B (since version 5.15).

\tkzTestInterLC(⟨O,A⟩)(⟨O′,B⟩)

The arguments are two pairs, which define a line and a circle (with a center and a point of the circle). If the line and the circle intersect, the test **\iftkzFlagLC** is true.

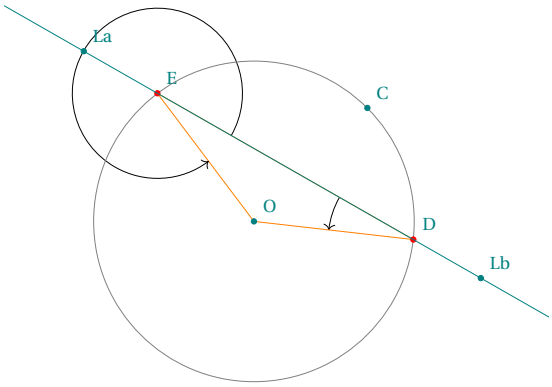
19.2.1. Testing a line-circle intersection: \tkzTestInterLC



```
\begin{tikzpicture}[scale=1]
  \tkzDefPoints{% x   y   name
                  3   /4   /I,
                  3   /2   /P,
                  0   /2   /La,
                  8   /3   /Lb}
\tkzDrawCircle(I,P)
\foreach \i in {1,...,3}{%
   \coordinate  (Lb) at (8,\i);
   \tkzDrawLine(La,Lb)
   \tkzTestInterLC(La,Lb)(I,P)
    \iftkzFlagLC
    \tkzInterLC(La,Lb)(I,P)
    \tkzGetPoints{a}{b}
    \tkzDrawPoints(a,b)
    \fi
   }
\end{tikzpicture}
```

19.2.2. Line-circle intersection

In the following example, the circle is drawn with two points, and the intersection of the straight line and the circle uses two pairs of points. We will compare the angles $\widehat{D,E,O}$ and $\widehat{E,D,O}$. These angles have opposite orientations. **tkzFirstPoint** is assigned to the point that forms the angle with the smallest measure (counterclockwise direction). The counterclockwise angle $\widehat{D,E,O}$ measures 360° minus the measure of $\widehat{O,E,D}$.

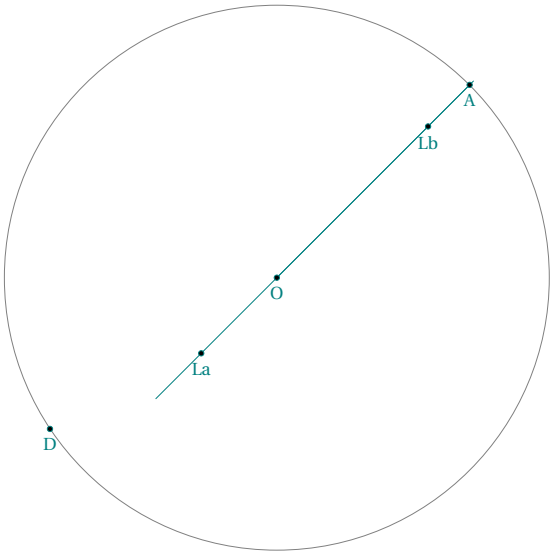



```
\begin{tikzpicture}[scale=.75]
 \tkzInit[xmax=5,ymax=4]
 \tkzDefPoint(1,1){O}
 \tkzDefPoint(-2,4){La}
 \tkzDefPoint(5,0){Lb}
 \tkzDefPoint(3,3){C}
 \tkzInterLC(La,Lb)(O,C)  \tkzGetPoints{D}{E}
 \tkzMarkAngle[->,size=1.5](E,D,O)
 \tkzDrawPolygons[new](O,D,E)
 \tkzMarkAngle[->,size=1.5](D,E,O)
 \tkzDrawCircle(O,C)
 \tkzDrawPoints[color=teal](O,La,Lb,C)
 \tkzDrawPoints[color=red](D,E)
 \tkzDrawLine(La,Lb)
 \tkzLabelPoints[above right](O,La,Lb,C,D,E)
\end{tikzpicture}
```

19.2.3. Line passing through the center

This case is special: the two points and the center are aligned, so the angles cannot be compared. Since version 5.15, the order is fixed: with **\tkzInterLC**(⟨La,Lb⟩)(⟨O,D⟩), **tkzSecondPoint** is the point located in the

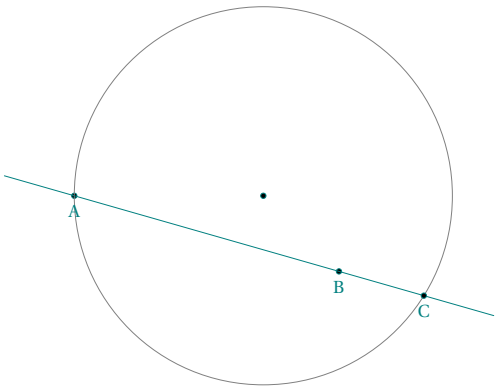
direction of the line, from La to Lb, and **tkzFirstPoint** is the other one. (Before 5.15, the order depended on rounding, and the results with and without the option **lua** could be different.)
You can also use the option **near**: **tkzFirstPoint** is then the point closest to the first point given for the line. Here we want A to be closest to Lb; **\tkzInterLC**(⟨Lb,La⟩)(⟨O,D⟩) without option gives the same point.




```
\begin{tikzpicture}
\tkzDefPoints{% x   y   name
              0    /1    /D,
              6    /0    /B,
              3    /3    /O,
              2    /2    /La,
              5    /5    /Lb}
  \tkzDrawCircle(O,D)
  \tkzDrawLine(La,Lb)
  \tkzInterLC[near](Lb,La)(O,D)
  \tkzGetFirstPoint{A}
  \tkzDrawSegments(O,A)
  \tkzDrawPoints(O,D,A,La,Lb)
  \tkzLabelPoints(O,D,A,La,Lb)
\end{tikzpicture}
```

19.2.4. Line-circle intersection with option common

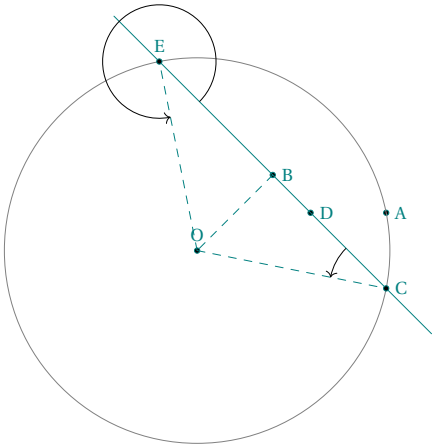
A special case that often occurs: a point of the line is on the circle, and we look for the other common point.




```
\begin{tikzpicture}[scale=.5]
 \tkzDefPoints{0/0/O,-5/0/A,2/-2/B,0/5/D}
 \tkzInterLC[common=A](B,A)(O,D)
 \tkzGetFirstPoint{C}
 \tkzDrawPoints(O,A,B)
 \tkzDrawCircle(O,A)
 \tkzDrawLine(A,C)
 \tkzDrawPoint(C)
 \tkzLabelPoints(A,B,C)
\end{tikzpicture}
```

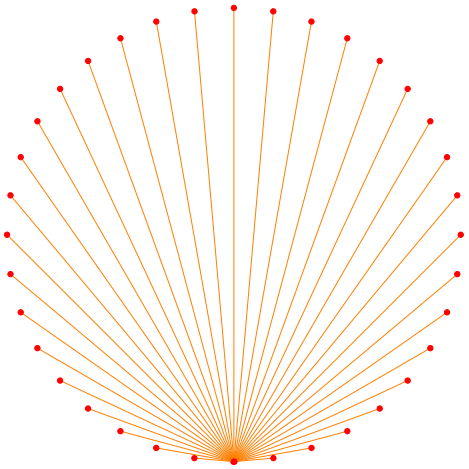
19.2.5. Line-circle intersection: order of the points

The idea is to compare the angles formed by the first point of the line, a resulting point and the center of the circle. The first point is the one that gives the smallest angle.
As you can see, $\widehat{BCO} < \widehat{BEO}$; in fact, $\widehat{BEO}$ is counterclockwise.




```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoints{0/0/O,5/1/A,2/2/B,3/1/D}
  \tkzInterLC[common=A](B,D)(O,A) \tkzGetPoints{C}{E}
  \tkzDrawPoints(O,A,B,D)
  \tkzDrawCircle(O,A) \tkzDrawLine(E,C)
  \tkzDrawSegments[dashed](B,O O,C)
  \tkzMarkAngle[->,size=1.5](B,C,O)
  \tkzDrawSegments[dashed](O,E)
  \tkzMarkAngle[->,size=1.5](B,E,O)
  \tkzDrawPoints(C,E)
  \tkzLabelPoints[above](O,E)
  \tkzLabelPoints[right](A,B,C,D)
\end{tikzpicture}
```

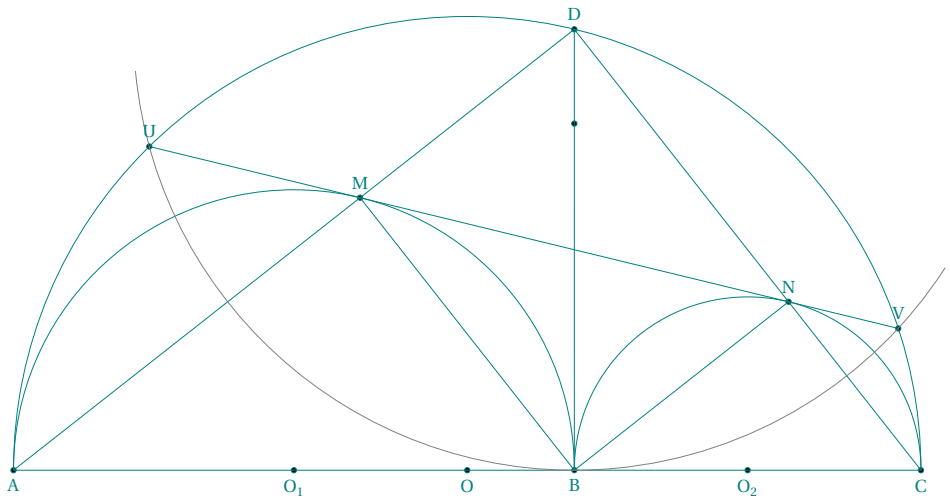
19.2.6. Example with \foreach



```
\begin{tikzpicture}[scale=3,rotate=180]
\tkzDefPoint(0,1){J}
\tkzDefPoint(0,0){O}
\foreach \i in {0,-5,-10,...,-90}{
 \tkzDefPoint({2.5*cos(\i*pi/180)},%
   {1+2.5*sin(\i*pi/180)}){P}
 \tkzInterLC[R](P,J)(O,1)\tkzGetPoints{N}{M}
 \tkzDrawSegment[color=orange](J,N)
 \tkzDrawPoints[red](N)}
\foreach \i in {-90,-95,...,-175,-180}{
 \tkzDefPoint({2.5*cos(\i*pi/180)},%
   {1+2.5*sin(\i*pi/180)}){P}
 \tkzInterLC[R](P,J)(O,1)\tkzGetPoints{N}{M}
 \tkzDrawSegment[color=orange](J,M)
 \tkzDrawPoints[red](M)}
\end{tikzpicture}
```

19.2.7. Line-circle intersection with option near

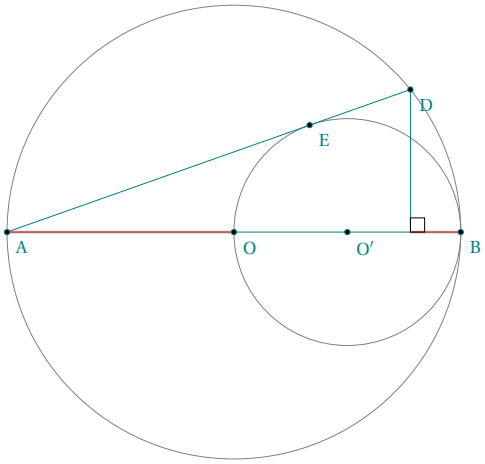
D is the point closest to b.



```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,12/0/C}
  \tkzDefGoldenRatio(A,C)                              \tkzGetPoint{B}
  \tkzDefMidPoint(A,C)                                 \tkzGetPoint{O}
  \tkzDefMidPoint(A,B)                                 \tkzGetPoint{O_1}
  \tkzDefMidPoint(B,C)                                 \tkzGetPoint{O_2}
  \tkzDefPointBy[rotation=center O_2 angle 90](C)      \tkzGetPoint{P}
  \tkzDefPointBy[rotation=center O_1 angle 90](B)      \tkzGetPoint{Q}
  \tkzDefPointBy[rotation=center B angle 90](C)        \tkzGetPoint{b}
  \tkzInterLC[near](b,B)(O,A)                          \tkzGetFirstPoint{D}
  \tkzInterCC(D,B)(O,C)                                \tkzGetPoints{V}{U}
  \tkzDefPointBy[projection=onto U--V](O_1)            \tkzGetPoint{M}
  \tkzDefPointBy[projection=onto U--V](O_2)            \tkzGetPoint{N}
  \tkzDrawPoints(A,B,C,O,O_1,O_2,D,U,V,M,N,b)
  \tkzDrawSemiCircles[teal](O,C O_1,B O_2,C)
  \tkzDrawSegments(A,C B,D U,V A,D C,D M,B B,N)
  \tkzDrawArc(D,U)(V)
  \tkzLabelPoints(A,B,C,O,O_1,O_2)
  \tkzLabelPoints[above](D,U,V,M,N)
\end{tikzpicture}
```

19.2.8. More complex example of a line-circle intersection

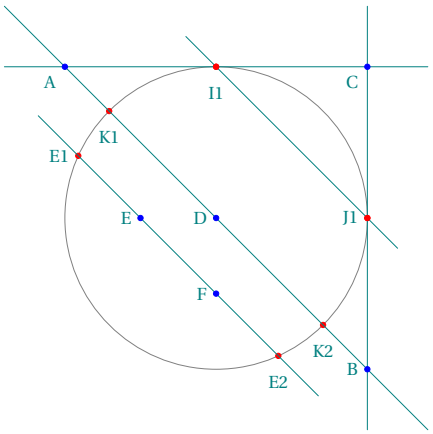
Figure from http://gogeometry.com/problem/p190_tangent_circle




```
\begin{tikzpicture}[scale=.75]
 \tkzDefPoint(0,0){A}
 \tkzDefPoint(8,0){B}
 \tkzDefMidPoint(A,B)                \tkzGetPoint{O}
 \tkzDefMidPoint(O,B)                \tkzGetPoint{O'}
 \tkzDefLine[tangent from=A](O',B)\tkzGetFirstPoint{E}
 \tkzInterLC(A,E)(O,B)               \tkzGetFirstPoint{D}
 \tkzDefPointBy[projection=onto A--B](D)
 \tkzGetPoint{F}
 \tkzDrawCircles(O,B O',B)
 \tkzDrawSegments(A,D A,B D,F)
 \tkzDrawSegments[color=red,line width=1pt,
     opacity=.4](A,O F,B)
 \tkzDrawPoints(A,B,O,O',E,D)
 \tkzMarkRightAngle(D,F,B)
 \tkzLabelPoints[below right](A,B,O,O',E,D)
\end{tikzpicture}
```

19.2.9. Circle defined by a center and a measure, and special cases

Let's look at some special cases, such as straight lines tangent to the circle.



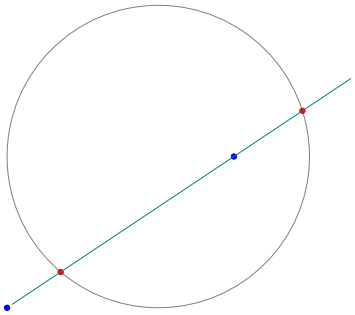

```
\begin{tikzpicture}[scale=.5]
 \tkzDefPoint(0,8){A}      \tkzDefPoint(8,0){B}
 \tkzDefPoint(8,8){C}      \tkzDefPoint(4,4){D}
 \tkzDefPoint(2,4){E}      \tkzDefPoint(4,2){F}
 \tkzDefPoint(8,4){G}
 \tkzInterLC(A,C)(D,G)     \tkzGetPoints{I1}{I2}
 \tkzInterLC(B,C)(D,G)     \tkzGetPoints{J1}{J2}
 \tkzInterLC[near](A,B)(D,G)  \tkzGetPoints{K1}{K2}
 \tkzInterLC(E,F)(D,G)     \tkzGetPoints{E1}{E2}
 \tkzDrawCircle(D,G)
 \tkzDrawPoints[color=red](I1,J1,K1,K2,E1,E2)
 \tkzDrawLines(A,B B,C A,C I2,J2 E1,E2)
 \tkzDrawPoints[color=blue](A,...,F)
 \tkzDrawPoints[color=red](I2,J2)
 \tkzLabelPoints[left](B,D,E,F)
 \tkzLabelPoints[below left](A,C)
 \tkzLabelPoints[below=4pt](I1,K1,K2,E2)
 \tkzLabelPoints[left](J1,E1)
\end{tikzpicture}
```

19.2.10. Calculation of radius

With **pgfmath** and **\pgfmathsetmacro**

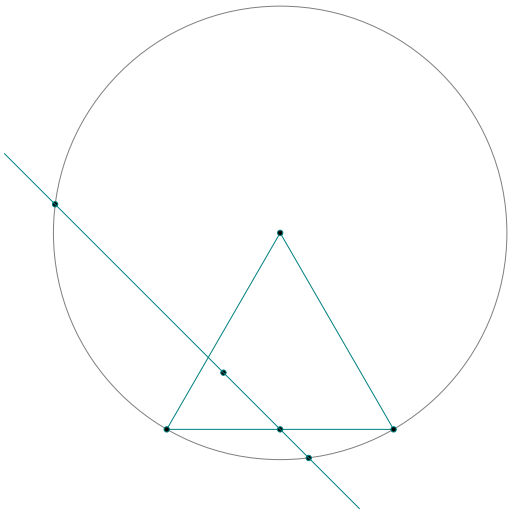
The radius may be the result of a calculation, done before the intersection macro rather than inside it. A length can be calculated in several ways. You can of course use the module **pgfmath** and the macro **\pgfmathsetmacro**. In some cases the results are not precise enough: the calculation 0.0002 ÷ 0.0001 gives 1.98 with pgfmath, while xfp gives 2.

With **xfp** and **\fpeval**:



```
  \begin{tikzpicture}
  \tkzDefPoint(2,2){A}
  \tkzDefPoint(5,4){B}
  \tkzDefPoint(4,4){O}
  \pgfmathsetmacro\tkzLen{\fpeval{0.0002/0.0001}}
 % or \edef\tkzLen{\fpeval{0.0002/0.0001}}
  \tkzInterLC[R](A,B)(O, \tkzLen)
  \tkzGetPoints{I}{J}
  \tkzDrawCircle(O,I)
  \tkzDrawPoints[color=blue](A,B)
  \tkzDrawPoints[color=red](I,J)
  \tkzDrawLine(I,J)
\end{tikzpicture}
```

19.2.11. Option with nodes



```
\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/0/A,4/0/B,1/1/D,2/0/E}
\tkzDefTriangle[equilateral](A,B)
\tkzGetPoint{C}
\tkzInterLC[with nodes](D,E)(C,A,B)
\tkzGetPoints{F}{G}
\tkzDrawCircle(C,A)
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints(A,...,G)
\tkzDrawLine(F,G)
\end{tikzpicture}
```

19.3. Intersection of two circles \tkzInterCC

The most frequent case is that of two circles defined by their center and a point, but as before the option R allows you to give the radius as a measure.

\tkzInterCC[⟨options⟩](⟨O,A⟩)(⟨O′,A′⟩) or (⟨O,r⟩)(⟨O′,r′⟩) or (⟨O,A,B⟩) (⟨O′,C,D⟩)

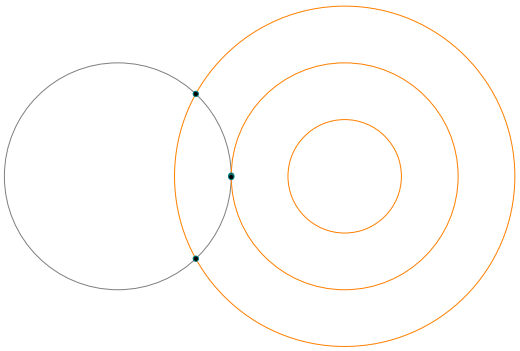
options	default	definition
N	N	OA and O′A′ are radii, O and O′ are the centers.
R	N	r and r′ are dimensions and measure the radii.
with nodes	N	in (A,A,C)(C,B,F) AC and BF give the radii.
common=pt		pt is common point; tkzFirstPoint gives the other point.

This macro defines the intersection points I and J of the two circles with centers O and O′. If the two circles have no common point, the macro ends with an error that is not handled. If the centers are O and O′ and the intersections are A and B, the angles $\widehat{O,A,O'}$ and $\widehat{O,B,O'}$ have opposite orientations; **tkzFirstPoint** is assigned to the point that forms the clockwise angle.

\tkzTestInterCC(⟨O,A⟩)(⟨O′,B⟩)

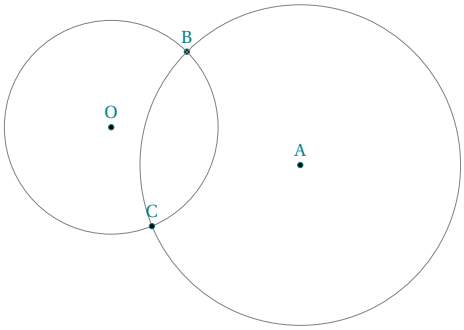
The arguments are two pairs, which define two circles (each with a center and a point of the circle). If the two circles intersect, the test **\iftkzFlagCC** is true.

19.3.1. Testing a circle-circle intersection: \tkzTestInterCC



```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoints{% x   y   name
                  0    /0    /A,
                  2    /0    /B,
                  4    /0    /I,
                  1    /0    /P}
\tkzDrawCircle(A,B)
\foreach \i in {1,...,3}{%
   \coordinate  (P) at (\i,0);
\tkzDrawCircle[new](I,P)
   \tkzTestInterCC(A,B)(I,P)
    \iftkzFlagCC
    \tkzInterCC(A,B)(I,P)  \tkzGetPoints{a}{b}
    \tkzDrawPoints(a,b)
    \fi}
  \end{tikzpicture}
```

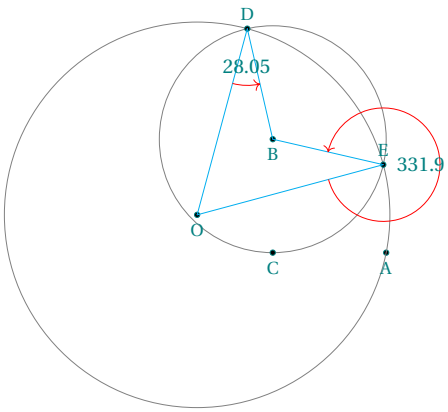
19.3.2. Circle-circle intersection with a common point




```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoints{0/0/O,5/-1/A,2/2/B}
  \tkzDrawPoints(O,A,B)
  \tkzDrawCircles(O,B A,B)
  \tkzInterCC[common=B](O,B)(A,B)
  \tkzGetFirstPoint{C}
  \tkzDrawPoint(C)
  \tkzLabelPoints[above](O,A,B,C)
\end{tikzpicture}
```

19.3.3. Circle-circle intersection: order of the points

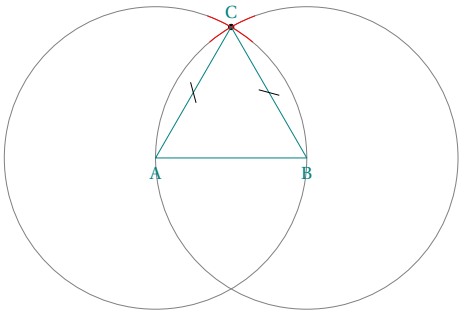
The idea is to compare the angles formed by the first center, a resulting point and the center of the second circle. The first point is the one that gives the smallest angle.
As you can see, $\widehat{\text{ODB}} < \widehat{\text{OBE}}$.




```
\begin{tikzpicture}[scale=.5]
  \pgfkeys{/pgf/number format/.cd,fixed relative,
     precision=4}
  \tkzDefPoints{0/0/O,5/-1/A,2/2/B,2/-1/C}
  \tkzDrawPoints(O,A,B)
  \tkzDrawCircles(O,A B,C)
  \tkzInterCC(O,A)(B,C)\tkzGetPoints{D}{E}
  \tkzDrawPoints(C,D,E)
  \tkzLabelPoints(O,A,B,C)
  \tkzLabelPoints[above](D,E)
  \tkzDrawSegments[cyan](D,O D,B)
  \tkzMarkAngle[red,->,size=1.5](O,D,B)
  \tkzFindAngle(O,D,B)   \tkzGetAngle{an}
  \tkzLabelAngle(O,D,B){$ \pgfmathprintnumber{\an}$}
  \tkzDrawSegments[cyan](E,O E,B)
  \tkzMarkAngle[red,->,size=1.5](O,E,B)
  \tkzFindAngle(O,E,B)   \tkzGetAngle{an}
  \tkzLabelAngle(O,E,B){$ \pgfmathprintnumber{\an}$}
\end{tikzpicture}
```

19.3.4. Construction of an equilateral triangle

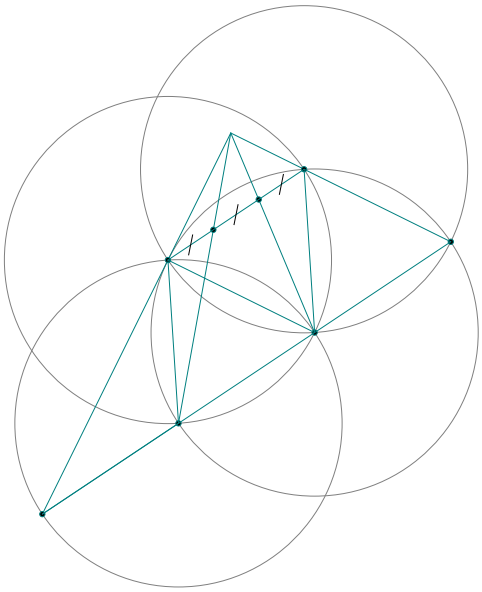
$\widehat{A,C,B}$ is a clockwise angle.




```
\begin{tikzpicture}[trim left=-1cm,scale=.5]
 \tkzDefPoint(1,1){A}
 \tkzDefPoint(5,1){B}
 \tkzInterCC(A,B)(B,A)\tkzGetPoints{C}{D}
 \tkzDrawPoint[color=black](C)
 \tkzDrawCircles(A,B B,A)
 \tkzCompass[color=red](A,C)
 \tkzCompass[color=red](B,C)
 \tkzDrawPolygon(A,B,C)
 \tkzMarkSegments[mark=s|](A,C B,C)
 \tkzLabelPoints[](A,B)
 \tkzLabelPoint[above](C){$C$}
\end{tikzpicture}
```

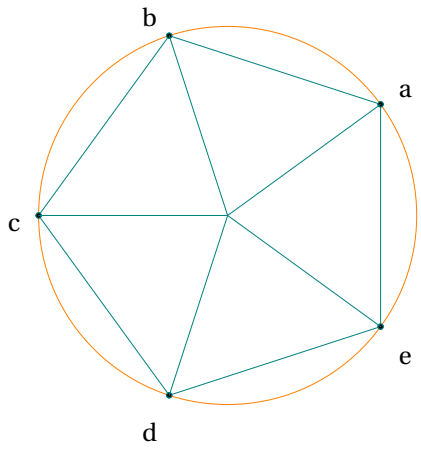
19.3.5. Segment trisection

The idea here is to divide a segment into three equal parts with a ruler and a compass.



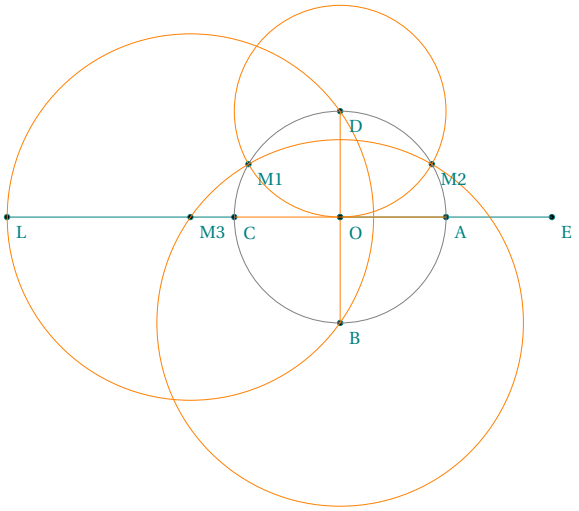
```
\begin{tikzpicture}[scale=.6]
 \tkzDefPoint(0,0){A}
 \tkzDefPoint(3,2){B}
 \tkzInterCC(A,B)(B,A)          \tkzGetSecondPoint{D}
 \tkzInterCC(D,B)(B,A)          \tkzGetPoints{A}{C}
 \tkzInterCC(D,B)(A,B)          \tkzGetPoints{E}{B}
 \tkzInterLC[common=D](C,D)(E,D)\tkzGetFirstPoint{F}
 \tkzInterLL(A,F)(B,C)          \tkzGetPoint{O}
 \tkzInterLL(O,D)(A,B)          \tkzGetPoint{H}
 \tkzInterLL(O,E)(A,B)          \tkzGetPoint{G}
 \tkzDrawCircles(D,E A,B B,A E,A)
 \tkzDrawSegments[](O,F O,B O,D O,E)
 \tkzDrawPoints(A,...,H)
 \tkzDrawSegments(A,B B,D A,D A,E E,F C,F B,C)
 \tkzMarkSegments[mark=s|](A,G G,H H,B)
\end{tikzpicture}
```

19.3.6. With the option with nodes




```
\begin{tikzpicture}[scale=.5]
 \tkzDefPoints{0/0/A,0/5/B,5/0/C}
 \tkzDefPoint(54:5){F}
 \tkzInterCC[with nodes](A,A,C)(C,B,F)
 \tkzGetPoints{a}{e}
 \tkzInterCC(A,C)(a,e) \tkzGetFirstPoint{b}
 \tkzInterCC(A,C)(b,a) \tkzGetFirstPoint{c}
 \tkzInterCC(A,C)(c,b) \tkzGetFirstPoint{d}
 \tkzDrawCircle[new](A,C)
 \tkzDrawPoints(a,b,c,d,e)
 \tkzDrawPolygon(a,b,c,d,e)
 \foreach \vertex/\num in {a/36,b/108,c/180,
                           d/252,e/324}{%
 \tkzDrawPoint(\vertex)
 \tkzLabelPoint[label=\num:$\vertex$](\vertex){}
 \tkzDrawSegment(A,\vertex)
 }
\end{tikzpicture}
```

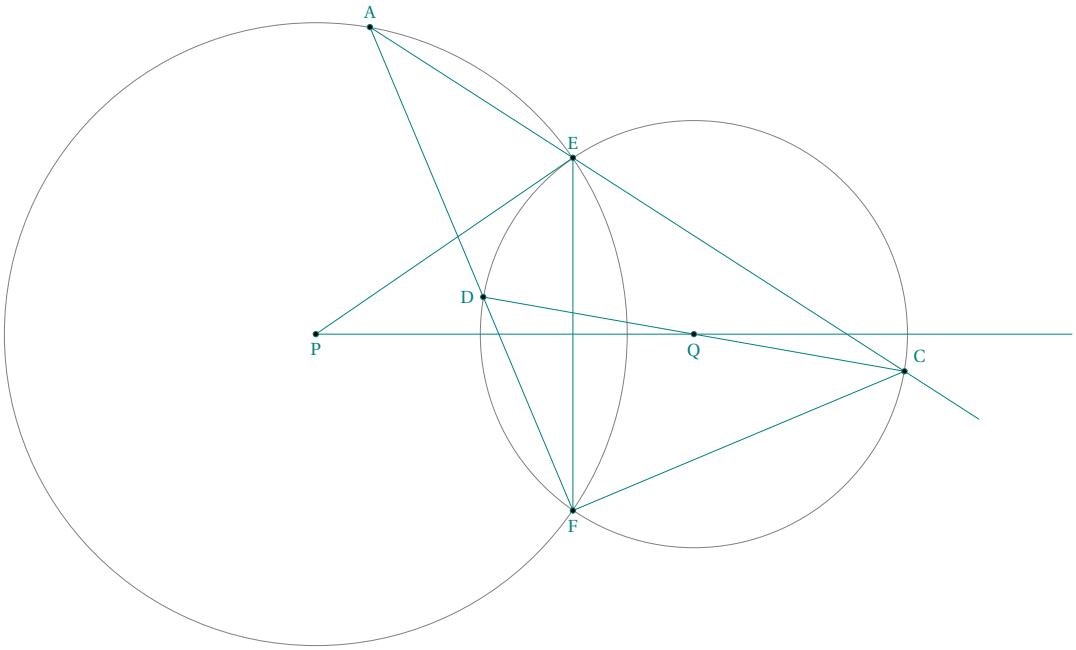
19.3.7. Mix of intersections




```
\begin{tikzpicture}[scale = .7]
  \tkzDefPoint(2,2){A}
  \tkzDefPoint(0,0){B}
  \tkzDefPoint(-2,2){C}
  \tkzDefPoint(0,4){D}
  \tkzDefPoint(4,2){E}
  \tkzCircumCenter(A,B,C)\tkzGetPoint{O}
  \tkzInterCC[R](O,2)(D,2)\tkzGetPoints{M1}{M2}
  \tkzInterCC(O,A)(D,O) \tkzGetPoints{1}{2}
  \tkzInterLC(A,E)(B,M1)\tkzGetSecondPoint{M3}
  \tkzInterLC(O,C)(M3,D)\tkzGetSecondPoint{L}
  \tkzDrawSegments(C,L)
  \tkzDrawPoints(A,B,C,D,E,M1,M2,M3,O,L)
  \tkzDrawSegments(O,E)
  \tkzDrawSegments[new](C,A D,B)
  \tkzDrawPoint(O)
  \tkzDrawCircles[new](M3,D B,M2 D,O)
  \tkzDrawCircle(O,A)
  \tkzLabelPoints[below right](A,B,C,D,E,M1,M2,M3,O,L)
\end{tikzpicture}
```

19.3.8. Altshiller-Court's theorem

The two lines joining the points of intersection of two orthogonal circles to a point of one of the circles meet the other circle in two diametrically opposite points (Altshiller-Court, p. 176).

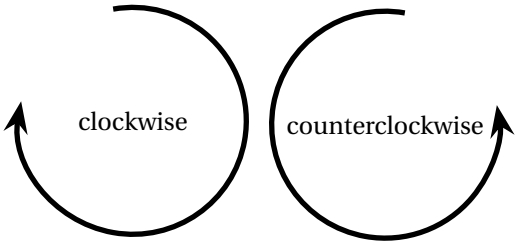



```
\begin{tikzpicture}
  \tkzDefPoints{0/0/P,5/0/Q,3/2/I}
  \tkzDefCircle[orthogonal from=P](Q,I)
  \tkzGetFirstPoint{E}
  \tkzDrawCircles(P,E Q,E)
  \tkzInterCC[common=E](P,E)(Q,E) \tkzGetFirstPoint{F}
  \tkzDefPointOnCircle[through =  center P angle 80 point E]
  \tkzGetPoint{A}
  \tkzInterLC[common=E](A,E)(Q,E)  \tkzGetFirstPoint{C}
  \tkzInterLL(A,F)(C,Q)  \tkzGetPoint{D}
  \tkzDrawLines[add=0 and 1](P,Q)
  \tkzDrawLines[add=0 and 2](A,E)
  \tkzDrawSegments(P,E E,F F,C A,F C,D)
  \tkzDrawPoints(P,Q,E,F,A,C,D)
  \tkzLabelPoints(P,Q,F)
  \tkzLabelPoints[above](E,A)
  \tkzLabelPoints[left](D)
  \tkzLabelPoints[above right](C)
\end{tikzpicture}
```

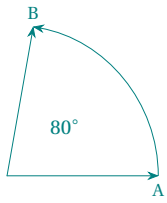
20. Angles

20.1. Definition and usage with tkz-euclide

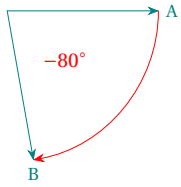
In Euclidean geometry, an angle is the figure formed by two rays, called the sides of the angle, sharing a common endpoint, called the vertex of the angle [Wikipedia]. With **tkz-euclide** a ray is defined by two points, so an angle is defined by three points, as in $\widehat{AOB}$; the vertex O is the second point. Their order matters, because the angle is assumed to be given counterclockwise. In trigonometry and mathematics in general, plane angles are conventionally measured counterclockwise, starting with 0° pointing directly to the right (or east), and 90° pointing straight up (or north) [Wikipedia]. Let us agree that an angle measured counterclockwise is positive.



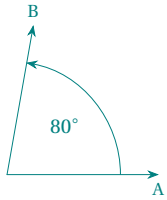
Angles are involved in several macros, such as `\tkzDefPoint`, `\tkzDefPointBy[rotation = …]`, `\tkzDrawArc` and `\tkzGetAngle`, described next. Except for the last one, all these macros accept negative angles.



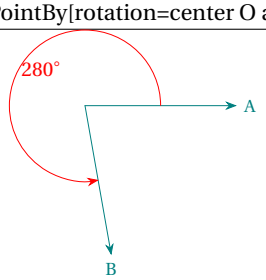
Rotation 80° from (O,A) to (O,B)
\tkzDefPointBy[rotation=center O angle 80]



Rotation −80° from (O,A) to (O,B)
\tkzDefPointBy[rotation=center O angle -80]



\tkzFindAngle(A,O,B) gives 80



\tkzFindAngle(A,O,B) gives 280°

As we can see, the −80° rotation defines a clockwise angle but the macro `\tkzFindAngle` recovers a counterclockwise angle.

20.2. Recovering an angle \tkzGetAngle

`\tkzGetAngle(⟨name of macro⟩)`

Assigns the value in degrees of an angle to a macro. The value is positive, between 0° and 360°. This macro retrieves `\tkzAngleResult` and stores it in a new macro.

arguments	example	explanation
`name of macro`	`\tkzGetAngle{ang}`	`\ang` contains the value of the angle.

This auxiliary macro retrieves the result of the next macro, `\tkzFindAngle`.

20.3. Angle formed by three points

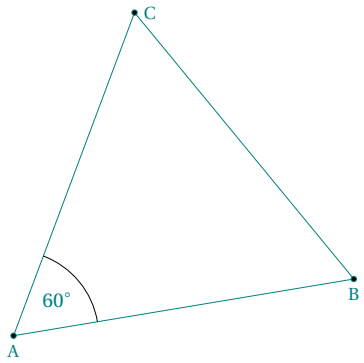
`\tkzFindAngle(⟨pt1,pt2,pt3⟩)`

The result is stored in a macro `\tkzAngleResult`.

arguments	example	explanation
`(pt1,pt2,pt3)`	`\tkzFindAngle(A,B,C)`	`\tkzAngleResult` gives the angle $(\overrightarrow{BA},\overrightarrow{BC})$

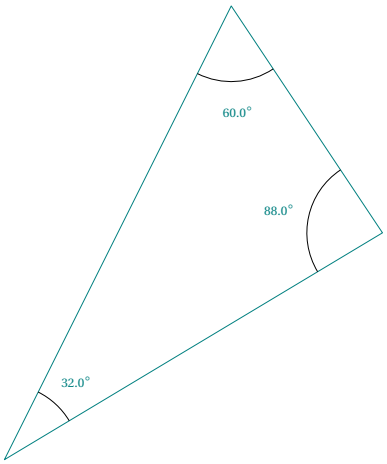
The measure is always positive, between 0° and 360°. With the usual conventions, a counterclockwise angle smaller than a straight angle always measures between 0° and 180°, while a clockwise angle smaller than a straight angle measures more than 180°. **\tkzGetAngle** retrieves the angle.

20.3.1. Verification of angle measurement




```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoint(-1,1){A}
  \tkzDefPoint(5,2){B}
  \tkzDefEquilateral(A,B)
  \tkzGetPoint{C}
  \tkzDrawPolygon(A,B,C)
  \tkzFindAngle(B,A,C) \tkzGetAngle{angleBAC}
  \edef\angleBAC{\fpeval{round(\angleBAC)}}
  \tkzDrawPoints(A,B,C)
  \tkzLabelPoints(A,B)
  \tkzLabelPoint[right](C){$C$}
  \tkzLabelAngle(B,A,C){\angleBAC$^\circ$}
  \tkzMarkAngle[size=1.5](B,A,C)
\end{tikzpicture}
```

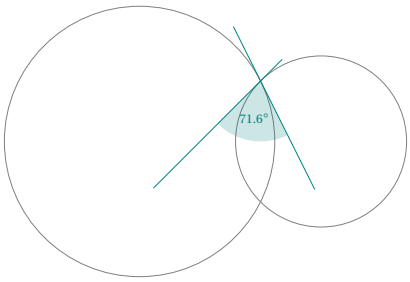
20.3.2. Determination of the three angles of a triangle




```
\begin{tikzpicture}
\tikzset{label angle style/.append style={pos=1.4}}
\tkzDefPoints{0/0/a,5/3/b,3/6/c}
\tkzDrawPolygon(a,b,c)
\tkzFindAngle(c,b,a)\tkzGetAngle{angleCBA}
\pgfmathparse{round(1+\angleCBA)}
\let\angleCBA\pgfmathresult
\tkzFindAngle(a,c,b)\tkzGetAngle{angleACB}
\pgfmathparse{round(\angleACB)}
\let\angleACB\pgfmathresult
\tkzFindAngle(b,a,c)\tkzGetAngle{angleBAC}
\pgfmathparse{round(\angleBAC)}
\let\angleBAC\pgfmathresult
\tkzMarkAngle(c,b,a)
\tkzLabelAngle(c,b,a){\tiny $\angleCBA^\circ$}
\tkzMarkAngle(a,c,b)
\tkzLabelAngle(a,c,b){\tiny $\angleACB^\circ$}
\tkzMarkAngle(b,a,c)
\tkzLabelAngle(b,a,c){\tiny $\angleBAC^\circ$}
\end{tikzpicture}
```

20.3.3. Angle between two circles

We look for the angle formed by the tangents at a point of intersection.




```
\begin{tikzpicture}[scale=.4]
\pgfkeys{/pgf/number format/.cd,%
         fixed,precision=1}
\tkzDefPoints{0/0/A,6/0/B,4/2/C}
\tkzDrawCircles(A,C B,C)
\tkzDefLine[tangent at=C](A) \tkzGetPoint{a}
\tkzDefPointsBy[symmetry = center C](a){d}
\tkzDefLine[tangent at=C](B) \tkzGetPoint{b}
\tkzDrawLines[add=1 and 4](a,C  C,b)
\tkzFillAngle[fill=teal,opacity=.2%
                          ,size=2](b,C,d)
\tkzFindAngle(b,C,d)\tkzGetAngle{bcd}
\tkzLabelAngle[pos=1.25](b,C,d){%
  \tiny $\pgfmathprintnumber{\bcd}^\circ$}
\end{tikzpicture}
```

20.4. Angle formed by a straight line with the horizontal axis \tkzFindSlopeAngle

More interesting than the previous one: the result is between −180° and +180°.

\tkzFindSlopeAngle(⟨A,B⟩)

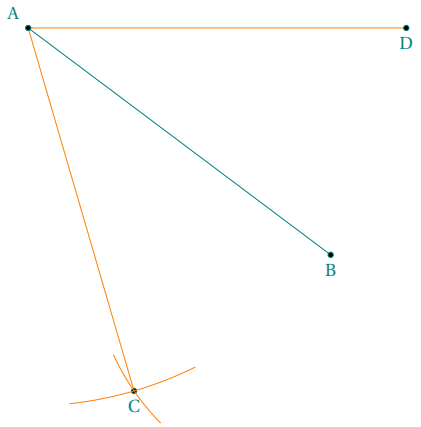
Determines the slope (as an angle) of the straight line (AB). The result is stored in the macro **\tkzAngleResult**.

arguments	example	explanation
(pt1,pt2)	\tkzFindSlopeAngle(A,B)	

\tkzGetAngle retrieves the result; if you don't need to keep it, you can use **\tkzAngleResult**.

20.4.1. How to use \tkzFindSlopeAngle

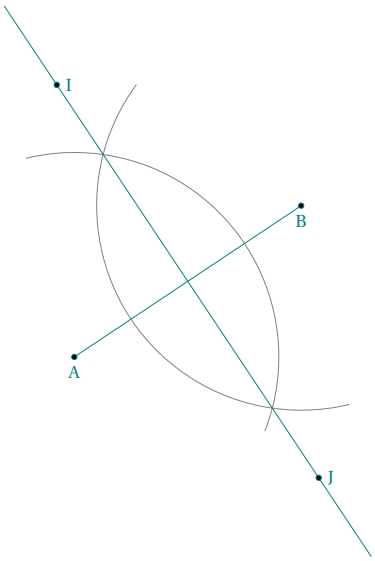
Here (AB) is the bisector of $\widehat{CAD}$, and the slope of (AD) is zero. We get the slope of (AB), then rotate twice.




```
\begin{tikzpicture}
 \tkzDefPoint(1,5){A} \tkzDefPoint(5,2){B}
 \tkzFindSlopeAngle(A,B)\tkzGetAngle{tkzang}
 \tkzDefPointBy[rotation= center A angle \tkzang ](B)
 \tkzGetPoint{C}
 \tkzDefPointBy[rotation= center A angle -\tkzang ](B)
 \tkzGetPoint{D}
 \tkzDrawSegment(A,B)
 \tkzDrawSegments[new](A,C A,D)
 \tkzDrawPoints(A,B,C,D)
 \tkzCompass[length=1](A,C)
 \tkzCompass[delta=10,brown](B,C)
 \tkzLabelPoints(B,C,D)
 \tkzLabelPoints[above left](A)
\end{tikzpicture}
```

20.4.2. Use of \tkzFindSlopeAngle and \tkzGetAngle

Here is another construction of a perpendicular bisector (mediator).



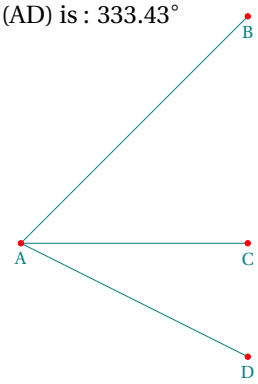

```
\begin{tikzpicture}
 \tkzInit
 \tkzDefPoint(0,0){A}          \tkzDefPoint(3,2){B}
 \tkzDefLine[mediator](A,B)    \tkzGetPoints{I}{J}
 \tkzCalcLength(A,B)           \tkzGetLength{dAB}
 \tkzFindSlopeAngle(A,B)       \tkzGetAngle{tkzangle}
 \begin{scope}[rotate=\tkzangle]
   \tkzSetUpArc[color=gray,line width=0.2pt,%
     /tkzcompass/delta=10]
   \tkzDrawArc[R,arc](B,3/4*\dAB)(120,240)
   \tkzDrawArc[R,arc](A,3/4*\dAB)(-45,60)
   \tkzDrawLine(I,J)           \tkzDrawSegment(A,B)
  \end{scope}
  \tkzDrawPoints(A,B,I,J)      \tkzLabelPoints(A,B)
   \tkzLabelPoints[right](I,J)
\end{tikzpicture}
```

20.4.3. Another use of \tkzFindSlopeAngle

The slope of (AB) is : 45°

The slope of (AC) is : 0°

The slope of (AD) is : 333.43°




```
\begin{tikzpicture}[scale=1.5]
  \tkzDefPoint(1,2){A}    \tkzDefPoint(3,4){B}
  \tkzDefPoint(3,2){C}    \tkzDefPoint(3,1){D}
  \tkzDrawSegments(A,B A,C A,D)
  \tkzDrawPoints[color=red](A,B,C,D)
  \tkzLabelPoints(A,B,C,D)
  \tkzFindSlopeAngle(A,B)\tkzGetAngle{SAB}
  \tkzFindSlopeAngle(A,C)\tkzGetAngle{SAC}
  \tkzFindSlopeAngle(A,D)\tkzGetAngle{SAD}
  \pgfkeys{/pgf/number format/.cd,fixed,precision=2}
  \tkzText(1,5){The slope of (AB) is :
     $\pgfmathprintnumber{\SAB}^\circ$}
  \tkzText(1,4.5){The slope of (AC) is :
     $\pgfmathprintnumber{\SAC}^\circ$}
  \tkzText(1,4){The slope of (AD) is :
     $\pgfmathprintnumber{\SAD}^\circ$}
\end{tikzpicture}
```

Part VI.

Drawing, filling and clipping

21. Drawing

tkz-euclide can draw five types of objects: points, lines or segments, circles, arcs and sectors.

21.1. Draw a point or some points

There are two possibilities: `\tkzDrawPoint` for a single point or `\tkzDrawPoints` for one or more points.

21.1.1. Drawing points \tkzDrawPoint

`\tkzDrawPoint[⟨local options⟩](⟨name⟩)`

arguments	default	definition
name of point	no default	Only one point name is accepted

The argument is required. The disc takes the color of the circle, but lighter; everything can be changed. The point is a node, so it is not affected when the drawing is scaled.

options	default	definition
TikZ options		all TikZ options are valid.
shape	circle	Possible **cross** or **cross out**
size	6	6× \pgflinewidth
color	black	the default color can be changed

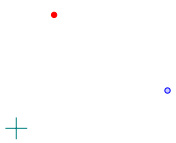
We can create other forms such as **cross**

By default, **point style** is defined like this:

```
\tikzset{point style/.style = {%
         draw         = black,
         inner sep    = 0pt,
         shape        = circle,
         minimum size = 3 pt,
         fill         = black
                              }
      }
```

21.1.2. Example of point drawings

Note that **scale** does not affect the shape of the points, which is normal. Most of the time, a single shape of point is enough; it can be defined at the beginning, either with a macro or in a configuration file.



```
\begin{tikzpicture}[scale=.5]
 \tkzDefPoint(1,3){A}
 \tkzDefPoint(4,1){B}
 \tkzDefPoint(0,0){O}
 \tkzDrawPoint[color=red](A)
 \tkzDrawPoint[fill=blue!20,draw=blue](B)
 \tkzDrawPoint[shape=cross,size=8pt,color=teal](O)
\end{tikzpicture}
```

You can draw several points at once, but this macro is a little slower than the previous one, and all the points get the same options.

\tkzDrawPoints[⟨local options⟩](⟨liste⟩)

arguments	default	definition
points list	no default	example \tkzDrawPoints(A,B,C)

options	default	definition
shape	circle	Possible cross or cross out
size	6	6× \pgflinewidth
color	black	the default color can be changed

Beware of the final s: forgetting it leads to cascading errors when you draw several points. The options are the same as for the previous macro.

21.1.3. Points with size, color and fill



```
\begin{tikzpicture}
\tkzDefPoints{1/3/A,4/1/B,0/0/C}
\tkzDrawPoints[size=3,color=red,fill=red!50](A,B,C)
\end{tikzpicture}
```

22. Drawing the lines

The following macros draw and label lines.

22.1. Draw a straight line

To draw a straight line, just give a pair of points. You can use the option add to extend the line (this option is due to Mark Wibrow, see the code below).
The style of a line is by default:

```
\tikzset{line style/.style = {%
  line width = 0.6pt,
  color      = black,
  style      = solid,
  add        = {.2} and  {.2}%
 }}
```

with

```
\tikzset{%
  add/.style args={#1 and #2}{
      to path={%
($(\tikztostart)!-#1!(\tikztotarget)$)--($(\tikztotarget)!-#2!(\tikztostart)$)%
 \tikztonodes}}}
```

You can modify this style with \tkzSetUpLine, see 44.1.

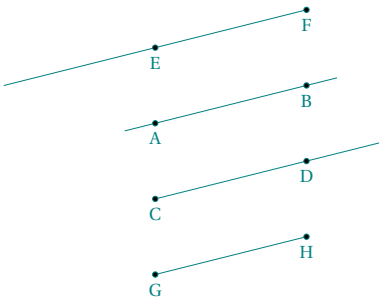
`\tkzDrawLine[⟨local options⟩](⟨pt1,pt2⟩)`

The arguments are a list of two or three points. As for a ray, you could create a style with **add**.

options	default	definition
TikZ options		all TikZ options are valid.
add	0.2 and 0.2	add = kl and kr, …
…	…	allows the segment to be extended to the left and right.

add defines the length of the line through the points pt1 and pt2. Both numbers are percentages. The Ti*k*Z styles are available for the drawings.

22.1.1. Examples with add



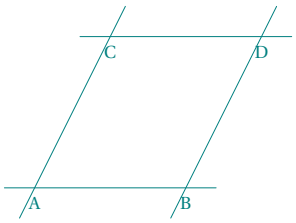

```
\begin{tikzpicture}
 \tkzInit[xmin=-2,xmax=3,ymin=-2.25,ymax=2.25]
 \tkzClip[space=.25]
 \tkzDefPoint(0,0){A} \tkzDefPoint(2,0.5){B}
 \tkzDefPoint(0,-1){C}\tkzDefPoint(2,-0.5){D}
 \tkzDefPoint(0,1){E} \tkzDefPoint(2,1.5){F}
 \tkzDefPoint(0,-2){G} \tkzDefPoint(2,-1.5){H}
 \tkzDrawLine(A,B)    \tkzDrawLine[add = 0 and .5](C,D)
 \tkzDrawLine[add = 1 and 0](E,F)
 \tkzDrawLine[add = 0 and 0](G,H)
 \tkzDrawPoints(A,B,C,D,E,F,G,H)
 \tkzLabelPoints(A,B,C,D,E,F,G,H)
\end{tikzpicture}
```

You can draw several lines at once, with the same options.

`\tkzDrawLines[⟨local options⟩](⟨pt1,pt2 pt3,pt4 ...⟩)`

The arguments are a list of pairs of points separated by spaces. The Ti*k*Z styles are available for the drawings.

22.1.2. Example with \tkzDrawLines




```
\begin{tikzpicture}
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(2,0){B}
  \tkzDefPoint(1,2){C}
  \tkzDefPoint(3,2){D}
  \tkzDrawLines(A,B C,D A,C B,D)
  \tkzLabelPoints(A,B,C,D)
\end{tikzpicture}
```

23. Drawing a segment

There is, of course, a macro to draw a segment.

23.1. Draw a segment \tkzDrawSegment

\tkzDrawSegment[⟨local options⟩](⟨pt1,pt2⟩)

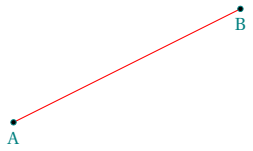
The arguments are a list of two points. The Ti*k*Z styles are available for the drawings.

argument	example	definition
(pt1,pt2)	(A,B)	draw the segment [A,B]

options	example	definition
TikZ options		all TikZ options are valid.
dim	no default	dim = {label,dim,option}, …
…	…	allows you to add dimensions to a figure.

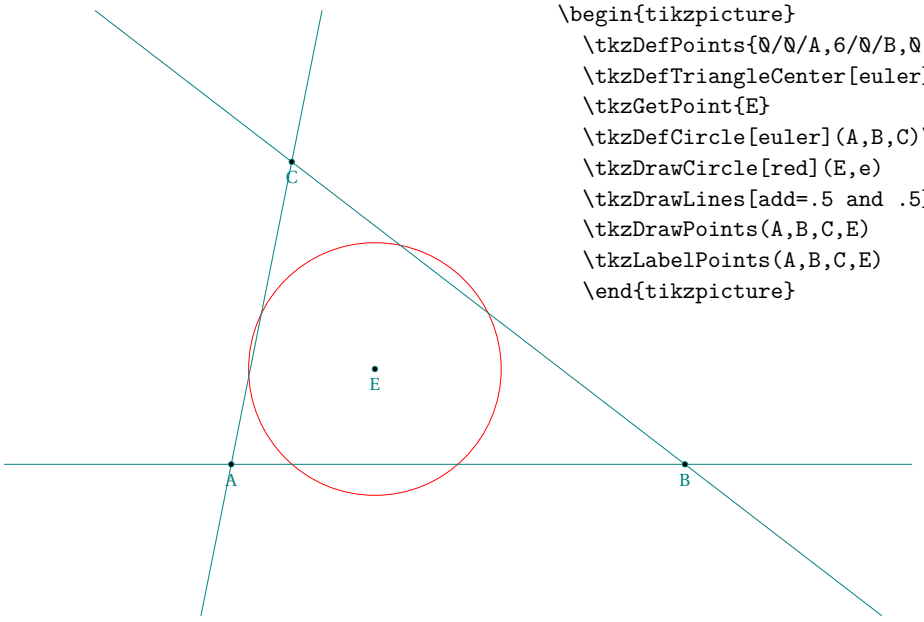
This is of course equivalent to `\draw (A)--(B);`. You can also use the option **add**.

23.1.1. Example with point references




```
\begin{tikzpicture}[scale=1.5]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(2,1){B}
  \tkzDrawSegment[color=red,thin](A,B)
  \tkzDrawPoints(A,B)
  \tkzLabelPoints(A,B)
\end{tikzpicture}
```

23.1.2. Extending a segment with option add

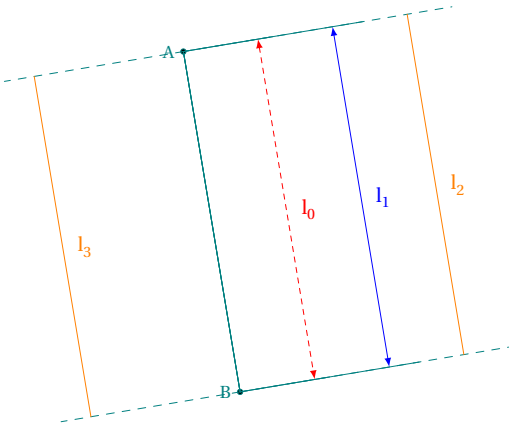



```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,6/0/B,0.8/4/C}
  \tkzDefTriangleCenter[euler](A,B,C)
  \tkzGetPoint{E}
  \tkzDefCircle[euler](A,B,C)\tkzGetPoints{E}{e}
  \tkzDrawCircle[red](E,e)
  \tkzDrawLines[add=.5 and .5](A,B A,C B,C)
  \tkzDrawPoints(A,B,C,E)
  \tkzLabelPoints(A,B,C,E)
  \end{tikzpicture}
```

23.1.3. Adding dimensions with option dim (code by Muzimuzhi Z)

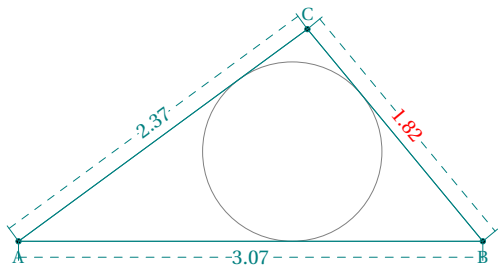
This code comes from an answer to a question on tex.stackexchange.com (change-color-and-style-of-dimension-lines-in-tkz-euclide). The code of **dim** is based on Ti*k*Z options, so you must give the units. You can now use two styles, `dim style` and `dim fence style`, in several ways. Look at the examples to see what you can do with these styles.

```
\tikzset{dim style/.append style={dashed}} % append if you want to keep precedent style.
or
\begin{scope}[ dim style/.append style={orange},
    dim fence style/.style={dashed}]
```



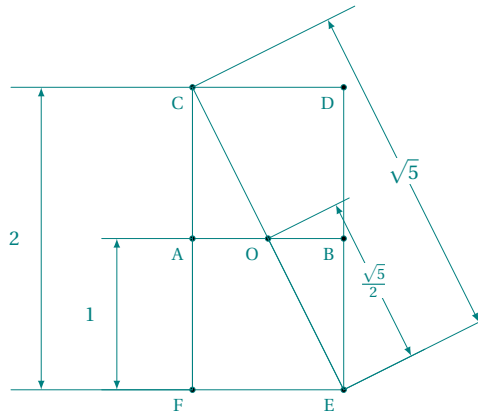

```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoints{0/3/A, 1/-3/B}
  \tkzDrawPoints(A,B)
  \tkzDrawSegment[dim={\(l_0\),1cm,right=2mm},
    dim style/.append style={red,
    dash pattern={on 2pt off 2pt}}](A,B)
  \tkzDrawSegment[dim={\(l_1\),2cm,right=2mm},
    dim style/.append style={blue}](A,B)
  \begin{scope}[ dim style/.style={orange},
      dim fence style/.style={dashed}]
    \tkzDrawSegment[dim={\(l_2\),3cm,right=2mm}](A,B)
    \tkzDrawSegment[dim={\(l_3\),-2cm,right=2mm}](A,B)
  \end{scope}
  \tkzLabelPoints[left](A,B)
\end{tikzpicture}
```

23.1.4. Adding dimensions with option dim: part I




```
\begin{tikzpicture}[scale=2]
\pgfkeys{/pgf/number format/.cd,fixed,precision=2}
\tkzDefPoint(0,0){A}
\tkzDefPoint(3.07,0){B}
\tkzInterCC[R](A,2.37)(B,1.82)
\tkzGetPoints{C}{C'}
\tkzDefCircle[in](A,B,C) \tkzGetPoints{G}{g}
\tkzDrawCircle(G,g)
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints(A,B,C)
\tkzCalcLength(A,B)\tkzGetLength{ABl}
\tkzCalcLength(B,C)\tkzGetLength{BCl}
\tkzCalcLength(A,C)\tkzGetLength{ACl}
\begin{scope}[dim style/.style={dashed,sloped,teal}]
  \tkzDrawSegment[dim={\pgfmathprintnumber\BCl,6pt,%
                                        text=red}](C,B)
  \tkzDrawSegment[dim={\pgfmathprintnumber\ACl,%
                                          6pt,}](A,C)
  \tkzDrawSegment[dim={\pgfmathprintnumber\ABl,%
                                        -6pt,}](A,B)
\end{scope}
\tkzLabelPoints(A,B) \tkzLabelPoints[above](C)
\end{tikzpicture}
```

23.1.5. Adding dimensions with option dim: part II



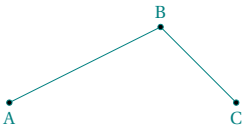

```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoints{0/0/O,-2/0/A,2/0/B,
                -2/4/C,2/4/D,2/-4/E,-2/-4/F}
  \tkzDrawPolygon(C,...,F)
  \tkzDrawSegments(A,B)
  \tkzDrawPoints(A,...,F,O)
  \tkzLabelPoints[below left](A,...,F,O)
  \tkzDrawSegment[dim={ $\sqrt{5}$,2cm,}](C,E)
  \tkzDrawSegment[dim={ $\frac{\sqrt{5}}{2}$,1cm,}](O,E)
  \tkzDrawSegment[dim={ $2$,2cm,left=8pt}](F,C)
  \tkzDrawSegment[dim={ $1$,1cm,left=8pt}](F,A)
\end{tikzpicture}
```

23.2. Drawing segments \tkzDrawSegments

If the options are the same, several segments can be drawn with one macro.

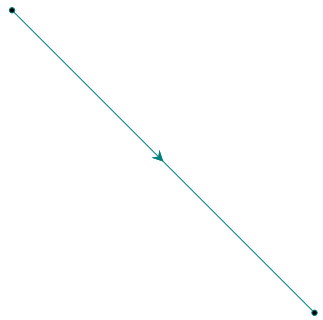
\tkzDrawSegments[⟨local options⟩](⟨pt1,pt2 pt3,pt4 ...⟩)

The arguments are a list of pairs of points. The Ti*kZ* styles are available for the drawings.




```
\begin{tikzpicture}
  \tkzInit[xmin=-1,xmax=3,ymin=-1,ymax=2]
  \tkzClip[space=1]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(2,1){B}
  \tkzDefPoint(3,0){C}
  \tkzDrawSegments(A,B B,C)
  \tkzDrawPoints(A,B,C)
  \tkzLabelPoints(A,C)
  \tkzLabelPoints[above](B)
\end{tikzpicture}
```

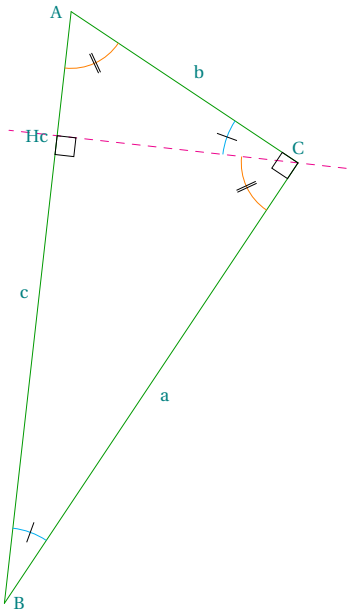
23.2.1. Placing an arrow on a segment



```
\begin{tikzpicture}
\tkzSetUpStyle[postaction=decorate,
    decoration={markings,
    mark=at position .5 with {\arrow[thick]{#1}}
      }]{myarrow}
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(4,-4){B}
  \tkzDrawSegments[myarrow=stealth](A,B)
  \tkzDrawPoints(A,B)
\end{tikzpicture}
```

23.3. Drawing the segments of a triangle

23.3.1. How to draw an altitude




```
\begin{tikzpicture}[rotate=-90]
\tkzDefPoint(0,1){A}
\tkzDefPoint(2,4){C}
\tkzDefPointWith[orthogonal normed,K=7](C,A)
\tkzGetPoint{B}
\tkzDefSpcTriangle[orthic,name=H](A,B,C){a,b,c}
\tkzDrawLine[dashed,color=magenta](C,Hc)
\tkzDrawSegment[green!60!black](A,C)
\tkzDrawSegment[green!60!black](C,B)
\tkzDrawSegment[green!60!black](B,A)
\tkzLabelPoint[left](A){$A$}
\tkzLabelPoint[right](B){$B$}
\tkzLabelPoint[above](C){$C$}
\tkzLabelPoint[left](Hc){$Hc$}
\tkzLabelSegment[auto](B,A){$c$}
\tkzLabelSegment[auto,swap](B,C){$a$}
\tkzLabelSegment[auto,swap](C,A){$b$}
\tkzMarkAngle[size=1,color=cyan,mark=|](C,B,A)
\tkzMarkAngle[size=1,color=cyan,mark=|](A,C,Hc)
\tkzMarkAngle[size=0.75,
              color=orange,mark=||](Hc,C,B)
\tkzMarkAngle[size=0.75,
              color=orange,mark=||](B,A,C)
\tkzMarkRightAngle(A,C,B)
\tkzMarkRightAngle(B,Hc,C)
\end{tikzpicture}
```

23.4. Drawing a polygon

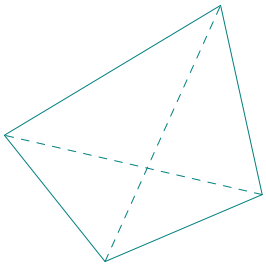
\tkzDrawPolygon[⟨local options⟩](⟨points list⟩)

Just give a list of points: the macro draws the polygon with the Ti*k*Z options given. You can replace (A,B,C,D,E) by (A,...,E) and (P_1,P_2,P_3,P_4,P_5) by (P_1,P...,P_5).

arguments	example	explanation
(⟨pt1,pt2,pt3,...⟩)	\tkzDrawPolygon[gray,dashed](A,B,C)	Drawing a triangle

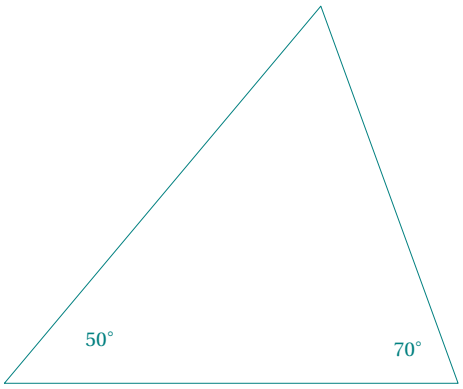
options	default	example
Options TikZ	...	\tkzDrawPolygon[red,line width=2pt](A,B,C)

23.4.1. \tkzDrawPolygon



```
\begin{tikzpicture} [rotate=18,scale=1]
 \tkzDefPoints{0/0/A,2.25/0.2/B,2.5/2.75/C,-0.75/2/D}
 \tkzDrawPolygon(A,B,C,D)
 \tkzDrawSegments[style=dashed](A,C B,D)
\end{tikzpicture}
```

23.4.2. Option two angles




```
\begin{tikzpicture}
\tkzDefPoint(0,0){A}
\tkzDefPoint(6,0){B}
\tkzDefTriangle[two angles = 50 and 70](A,B) \tkzGetPoint{C}
\tkzDrawPolygon(A,B,C)
\tkzLabelAngle[pos=1.4](B,A,C){$50^\circ$}
\tkzLabelAngle[pos=0.8](C,B,A){$70^\circ$}
\end{tikzpicture}
```

23.4.3. Style of line



```
\begin{tikzpicture}[scale=.6]
\tkzSetUpLine[line width=5mm,color=teal]
\tkzDefPoint(0,0){O}
\foreach \i in {0,...,5}{%
 \tkzDefPoint({30+60*\i}:4){p\i}}
\tkzDefMidPoint(p1,p3) \tkzGetPoint{m1}
\tkzDefMidPoint(p3,p5) \tkzGetPoint{m3}
\tkzDefMidPoint(p5,p1) \tkzGetPoint{m5}
\tkzDrawPolygon[line join=round](p1,p3,p5)
\tkzDrawPolygon[teal!80,
line join=round](p0,p2,p4)
\tkzDrawSegments(m1,p3 m3,p5 m5,p1)
\tkzDefCircle[R](O,4.8)\tkzGetPoint{o}
\tkzDrawCircle[teal](O,o)
\end{tikzpicture}
```

23.5. Drawing a polygonal chain

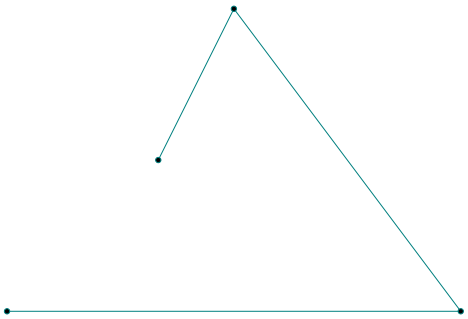
\tkzDrawPolySeg[⟨local options⟩](⟨points list⟩)

Just give a list of points: the macro draws the polygonal chain with the Ti*k*Z options given.

arguments	example	explanation
(⟨pt1,pt2,pt3,...⟩)	\tkzDrawPolySeg[gray,dashed](A,B,C)	Drawing a triangle

options	default	example
Options TikZ	...	\tkzDrawPolySeg[red,line width=2pt](A,B,C)

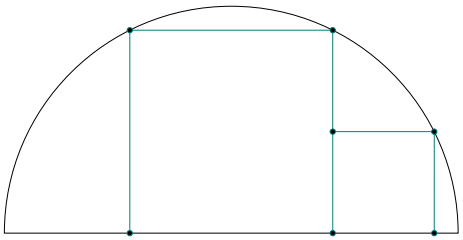
23.5.1. Polygonal chain



```
\begin{tikzpicture}
 \tkzDefPoints{0/0/A,6/0/B,3/4/C,2/2/D}
 \tkzDrawPolySeg(A,...,D)
 \tkzDrawPoints(A,...,D)
\end{tikzpicture}
```

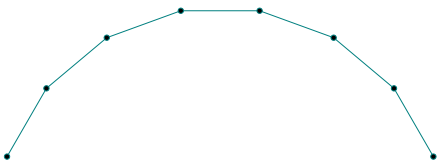
23.5.2. The idea is to inscribe two squares in a semi-circle.

A Sangaku! The problem is to prove that two squares can be inscribed in a half-disc, and to determine the lengths of their sides from the radius.



```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoints{0/0/A,8/0/B,4/0/I}
  \tkzDefSquare(A,B)     \tkzGetPoints{C}{D}
  \tkzInterLC(I,C)(I,B) \tkzGetPoints{E'}{E}
  \tkzInterLC(I,D)(I,B) \tkzGetPoints{F'}{F}
  \tkzDefPointsBy[projection=onto A--B](E,F){H,G}
  \tkzDefPointsBy[symmetry = center H](I){J}
  \tkzDefSquare(H,J)      \tkzGetPoints{K}{L}
  \tkzDrawSector(I,B)(A)
  \tkzDrawPolySeg(H,E,F,G)
  \tkzDrawPolySeg(J,K,L)
  \tkzDrawPoints(E,G,H,F,J,K,L)
\end{tikzpicture}
```

23.5.3. Polygonal chain: index notation



```
\begin{tikzpicture}
\foreach \pt in {1,2,...,8} {%
\tkzDefPoint(\pt*20:3){P_\pt}}
\tkzDrawPolySeg(P_1,P_...,P_8)
\tkzDrawPoints(P_1,P_...,P_8)
\end{tikzpicture}
```

24. Draw a circle with \tkzDrawCircle

24.1. Draw one circle

\tkzDrawCircle[⟨local options⟩](⟨A,B⟩)

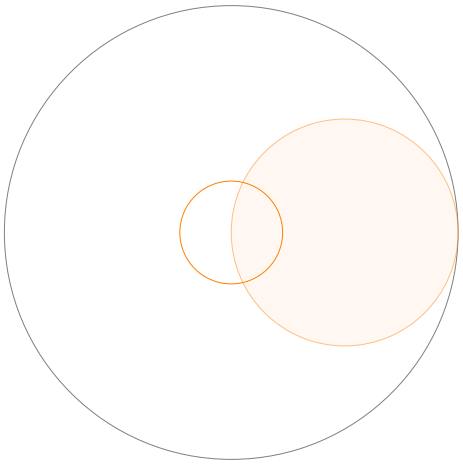
Note that only two points are needed to define a radius. The option R lets you give the radius directly as a measure.

arguments	example	explanation
(⟨pt1,pt2⟩)	(⟨A,B⟩)	A center through B

Of course, you can add all the Ti*k*Z styles for the drawings.

24.1.1. Circles and styles, draw a circle and color the disc

We'll see that it's possible to color a disc while drawing the circle.



```
\begin{tikzpicture}
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(3,0){A}
 % circle with center O and passing through A
  \tkzDrawCircle(O,A)
 % diameter circle $[OA]$
 \tkzDefCircle[diameter](O,A) \tkzGetPoint{I}
 \tkzDrawCircle[new,fill=orange!10,opacity=.5](I,A)
 % circle with center O and radius = exp(1) cm
  \edef\rayon{\fpeval{0.25*exp(1)}}
  \tkzDefCircle[R](O,\rayon) \tkzGetPoint{o}
   \tkzDrawCircle[color=orange](O,o)
\end{tikzpicture}
```

24.2. Drawing circles

`\tkzDrawCircles[⟨local options⟩](⟨A,B C,D …⟩)`

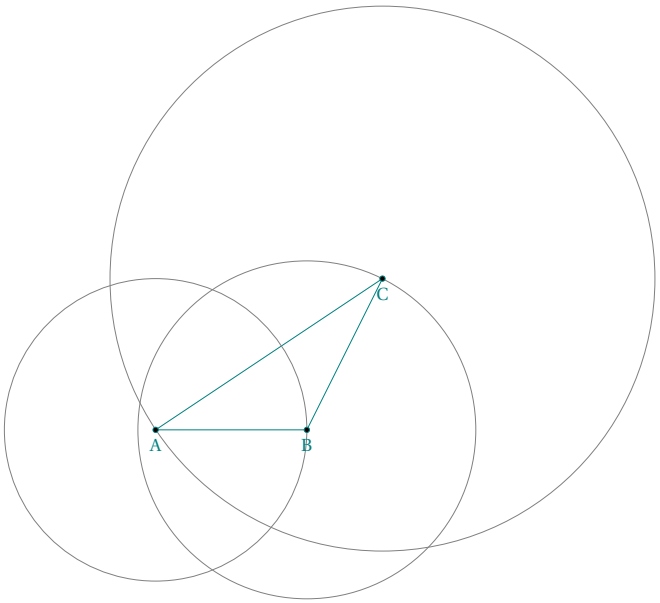
Note that the arguments are lists of pairs of points. The circles that can be drawn are the same as with the previous macro. The option `R` lets you give the radius directly as a measure.

arguments	example	explanation
(⟨pt1,pt2 pt3,pt4 ...⟩)	(⟨A,B C,D⟩)	List of two points

options	default	definition
through	through	circle with two points defining a radius

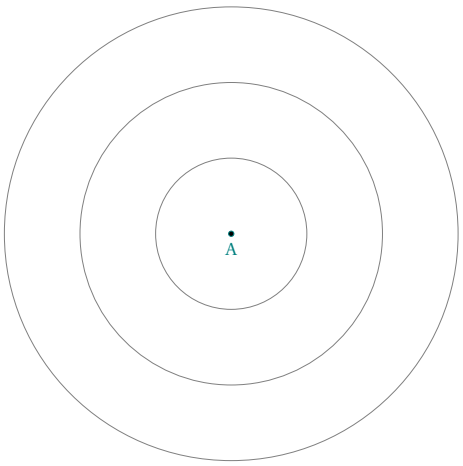
You do not need to use the default option **through**. Of course, you can add all the Ti*k*Z styles for the drawings.

24.2.1. Circles defined by a triangle.



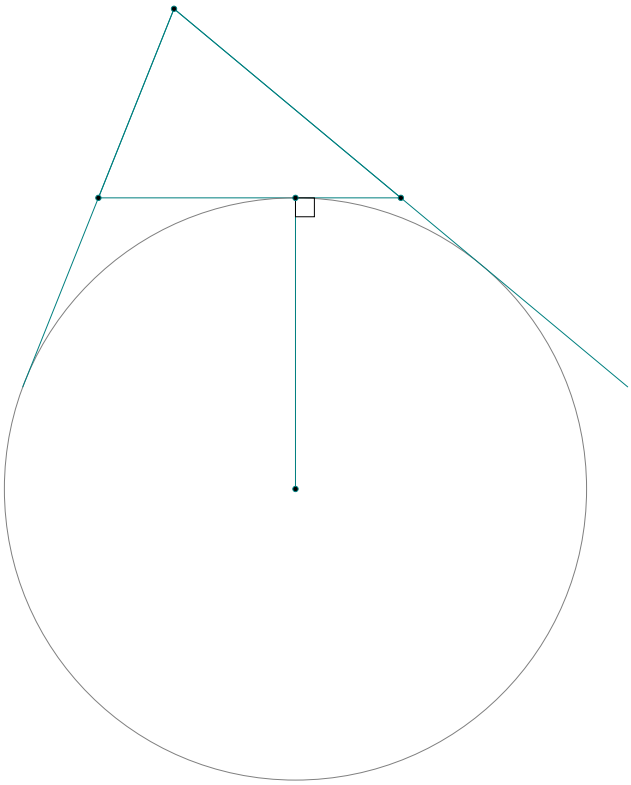

```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,2/0/B,3/2/C}
  \tkzDrawPolygon(A,B,C)
  \tkzDrawCircles(A,B B,C C,A)
  \tkzDrawPoints(A,B,C)
  \tkzLabelPoints(A,B,C)
\end{tikzpicture}
```

24.2.2. Concentric circles.




```
\begin{tikzpicture}
   \tkzDefPoints{0/0/A,1/0/a,2/0/b,3/0/c}
   \tkzDrawCircles(A,a A,b A,c)
   \tkzDrawPoint(A)
   \tkzLabelPoints(A)
\end{tikzpicture}
```

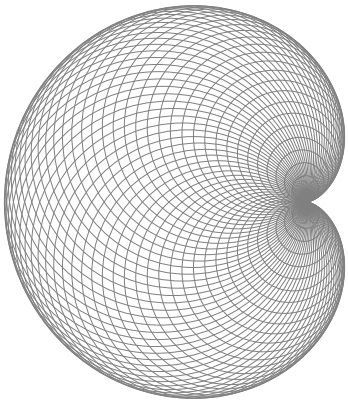
24.2.3. Exinscribed circles.



```
\begin{tikzpicture}[scale=1]
\tkzDefPoints{0/0/A,4/0/B,1/2.5/C}
\tkzDrawPolygon(A,B,C)
\tkzDefCircle[ex](B,C,A)
\tkzGetPoint{J_c} \tkzGetSecondPoint{T_c}
\tkzDrawCircle(J_c,T_c)
\tkzDrawLines[add=0 and 1](C,A C,B)
\tkzDrawSegment(J_c,T_c)
\tkzMarkRightAngle(J_c,T_c,B)
\tkzDrawPoints(A,B,C,J_c,T_c)
\end{tikzpicture}
```

24.2.4. Cardioid

Based on an idea of O. Reboux, made with pst-eucl (a PSTricks module) by D. Rodriguez.
Its name comes from the Greek *kardia* (heart), in reference to its shape; it was given by Johan Castillon (Wikipedia).



```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2,0){A}
  \foreach \ang in {5,10,...,360}{%
     \tkzDefPoint(\ang:2){M}
     \tkzDrawCircle(M,A)
   }
\end{tikzpicture}
```

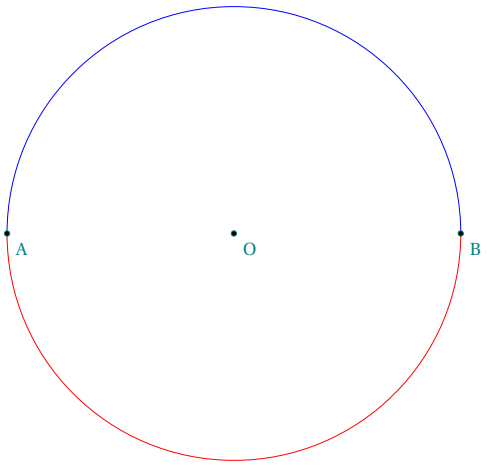
24.3. Drawing a semicircle

\tkzDrawSemiCircle[⟨local options⟩](⟨O,A⟩)

arguments	example	explanation
(⟨pt1,pt2⟩)	(⟨O,A⟩)	OA= radius

O is the center and A an endpoint of the semicircle.

24.3.1. Use of \tkzDrawSemiCircle



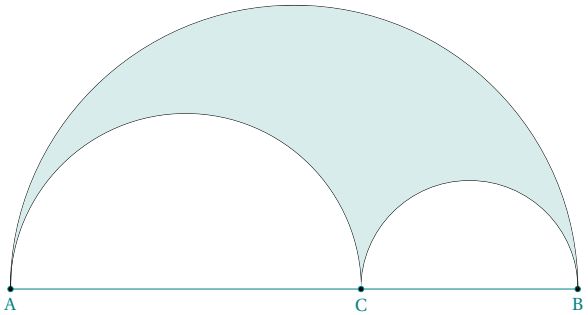

```
\begin{tikzpicture}
   \tkzDefPoint(0,0){A} \tkzDefPoint(6,0){B}
   \tkzDefMidPoint(A,B)  \tkzGetPoint{O}
   \tkzDrawSemiCircle[blue](O,B)
   \tkzDrawSemiCircle[red](O,A)
   \tkzDrawPoints(O,A,B)
   \tkzLabelPoints[below right](O,A,B)
 \end{tikzpicture}
```

24.4. Drawing semicircles

\tkzDrawSemiCircles[⟨local options⟩](⟨A,B C,D …⟩)

arguments	example	explanation
(⟨pt1,pt2 pt3,pt4 ...⟩)	(⟨A,B C,D⟩)	List of two points

24.4.1. Use of \tkzDrawSemiCircles: Golden arbelos



```
\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/0/A,10/0/B}
\tkzDefGoldenRatio(A,B) \tkzGetPoint{C}
\tkzDefMidPoint(A,B)                    \tkzGetPoint{O_0}
\tkzDefMidPoint(A,C)                    \tkzGetPoint{O_1}
\tkzDefMidPoint(C,B)                    \tkzGetPoint{O_2}
\tkzLabelPoints(A,B,C)
\tkzDrawSegment(A,B)
\tkzDrawPoints(A,B,C)
\begin{scope}[local bounding box = graph]
  \tkzDrawSemiCircles[color=black](O_0,B)
\end{scope}
\useasboundingbox (graph.south west) rectangle (graph.north east);
\tkzClipCircle[out](O_1,C)\tkzClipCircle[out](O_2,B)
\tkzDrawSemiCircles[draw=none,fill=teal!15](O_0,B)
\tkzDrawSemiCircles[color=black](O_1,C O_2,B)
\end{tikzpicture}
```

25. Draw an ellipse with \tkzDrawEllipse

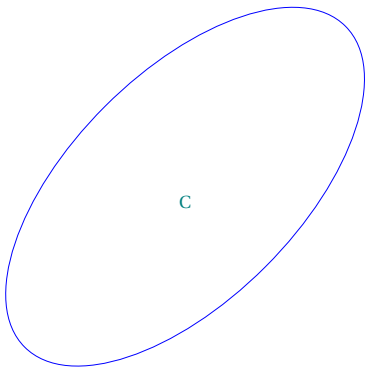
25.1. Draw an ellipse

\tkzDrawEllipse[⟨local options⟩](⟨C,a,b,An⟩)

arguments	example	explanation
(⟨C,a,b,An⟩)	(⟨C,4,2,45⟩)	C center; 4 and 2 lengths of half-axis 45 slope of main axis

Of course, you can add all the Ti*k*Z styles for the drawings.

25.1.1. Example of drawing an ellipse




```
\begin{tikzpicture}[scale=.75]
   \tkzDefPoint(0,4){C}
   \tkzDrawEllipse[blue](C,4,2,45)
   \tkzLabelPoints(C)
\end{tikzpicture}
```

26. Drawing arcs

26.1. The macro \tkzDrawArc

\tkzDrawArc[⟨local options⟩](⟨O,…⟩)(⟨…⟩)

This macro draws an arc with center O; the arguments depend on the options. We need a starting point and an

end point. Either the starting point is given, which is the simplest, or the radius of the arc is given; in the latter case, two angles are needed. The angles can be given directly, or with points which, together with the center, determine them. The angles are in degrees.

options	default	definition
towards	towards	O is the center and the arc from A to (OB)
rotate	towards	the arc starts from A and the angle determines its length
R	towards	We give the radius and two angles
R with nodes	towards	We give the radius and two points
angles	towards	We give the radius and two points
delta	0	angle added on each side
reverse	false	inversion of the arc's path, interesting to inverse arrow

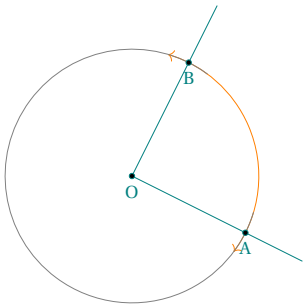
Of course, you can add all the Ti*k*Z styles for the drawings.

options	arguments	example
towards	(⟨pt,pt⟩)(⟨pt⟩)	\tkzDrawArc[delta=10](O,A)(B)
rotate	(⟨pt,pt⟩)(⟨an⟩)	\tkzDrawArc[rotate,color=red](O,A)(90)
R	(⟨pt,r⟩)(⟨an,an⟩)	\tkzDrawArc[R](O,2)(30,90)
R with nodes	(⟨pt,r⟩)(⟨pt,pt⟩)	\tkzDrawArc[R with nodes](O,2)(A,B)
angles	(⟨pt,pt⟩)(⟨an,an⟩)	\tkzDrawArc[angles](O,A)(0,90)

Here are a few examples:

26.1.1. Option towards: arc from A to B

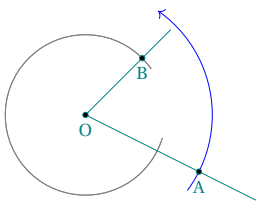
You don't need to write **towards**. In this first example, the arc starts from A and goes to B. The arc from B to A is different: arcs are always drawn counterclockwise.




```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2,-1){A}
  \tkzDefPointBy[rotation= center O angle 90](A)
  \tkzGetPoint{B}
  \tkzDrawArc[color=orange,<->](O,A)(B)
  \tkzDrawArc(O,B)(A)
  \tkzDrawLines[add = 0 and .5](O,A O,B)
  \tkzDrawPoints(O,A,B)
  \tkzLabelPoints[below](O,A,B)
\end{tikzpicture}
```

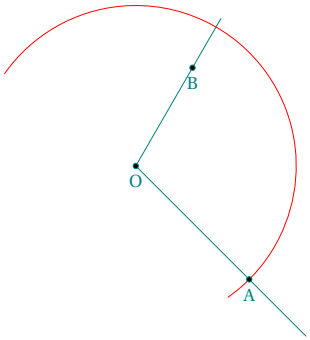
26.1.2. Option towards: arc stopping on a line

In this one, the arc starts from A but stops on the line (OB).



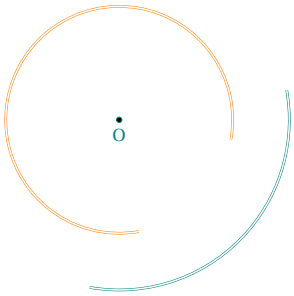

```
\begin{tikzpicture}[scale=0.75]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2,-1){A}
  \tkzDefPoint(1,1){B}
  \tkzDrawArc[color=blue,->](O,A)(B)
  \tkzDrawArc[color=gray](O,B)(A)
  \tkzDrawArc(O,B)(A)
  \tkzDrawLines[add = 0 and .5](O,A O,B)
  \tkzDrawPoints(O,A,B)
  \tkzLabelPoints[below](O,A,B)
\end{tikzpicture}
```

26.1.3. Option rotate



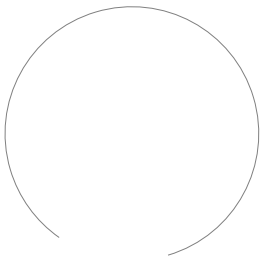

```
\begin{tikzpicture}[scale=0.75]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2,-2){A}
  \tkzDefPoint(60:2){B}
  \tkzDrawLines[add = 0 and .5](O,A O,B)
  \tkzDrawArc[rotate,color=red](O,A)(180)
  \tkzDrawPoints(O,A,B)
  \tkzLabelPoints[below](O,A,B)
\end{tikzpicture}
```

26.1.4. Option R




```
\begin{tikzpicture}[scale=0.75]
  \tkzDefPoints{0/0/O}
  \tkzSetUpCompass[<->]
  \tkzDrawArc[R,color=teal,double](O,3)(270,360)
  \tkzDrawArc[R,color=orange,double](O,2)(0,270)
  \tkzDrawPoint(O)
  \tkzLabelPoint[below](O){$O$}
\end{tikzpicture}
```

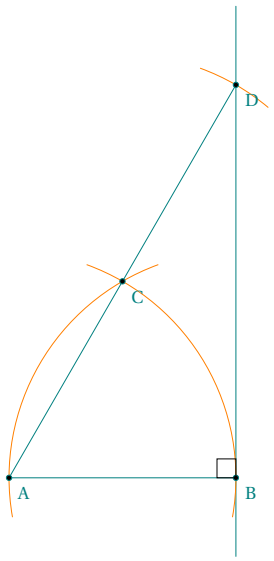
26.1.5. Option R with nodes



```
\begin{tikzpicture}[scale=0.75]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2,-1){A}
  \tkzDefPoint(1,1){B}
  \tkzCalcLength(B,A)\tkzGetLength{radius}
  \tkzDrawArc[R with nodes](B,\radius)(A,O)
\end{tikzpicture}
```

26.1.6. Option delta

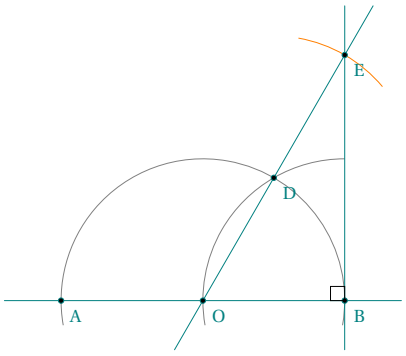
As with **\tkzCompass**, this option extends the arc on both sides; **delta** is an angle in degrees. It can also be set for all the arcs with **\tkzSetUpArc**.




```
\begin{tikzpicture}
 \tkzDefPoint(0,0){A}
 \tkzDefPoint(3,0){B}
 \tkzDefPointBy[rotation= center A angle 60](B)
 \tkzGetPoint{C}
 \begin{scope}% style only local
   \tkzDefPointBy[symmetry= center C](A)
   \tkzGetPoint{D}
   \tkzDrawSegments(A,B A,D)
   \tkzDrawLine(B,D)
   \tkzSetUpCompass[color=orange]
   \tkzDrawArc[orange,delta=10](A,B)(C)
   \tkzDrawArc[orange,delta=10](B,C)(A)
   \tkzDrawArc[orange,delta=10](C,D)(D)
 \end{scope}

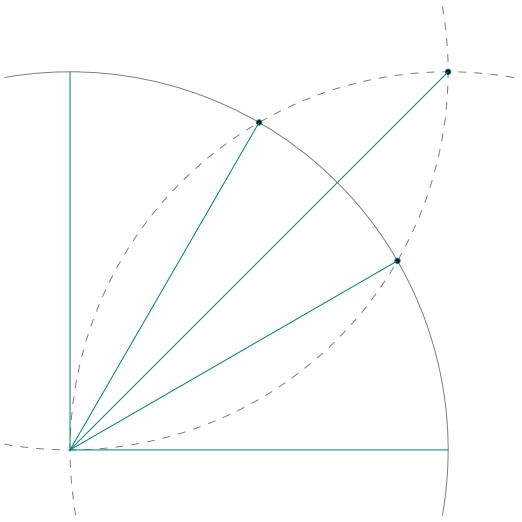
 \tkzDrawPoints(A,B,C,D)
 \tkzLabelPoints[below right](A,B,C,D)
 \tkzMarkRightAngle(D,B,A)
\end{tikzpicture}
```

26.1.7. Option angles: example 1



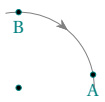

```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(5,0){B}
  \tkzDefPoint(2.5,0){O}
  \tkzDefPointBy[rotation=center O angle 60](B)
  \tkzGetPoint{D}
  \tkzDefPointBy[symmetry=center D](O)
  \tkzGetPoint{E}
  \begin{scope}
    \tkzDrawArc[angles](O,B)(0,180)
    \tkzDrawArc[angles,](B,O)(100,180)
    \tkzCompass[delta=20](D,E)
    \tkzDrawLines(A,B O,E B,E)
    \tkzDrawPoints(A,B,O,D,E)
  \end{scope}
  \tkzLabelPoints[below right](A,B,O,D,E)
  \tkzMarkRightAngle(O,B,E)
\end{tikzpicture}
```

26.1.8. Option angles: example 2



```
\begin{tikzpicture}
 \tkzDefPoint(0,0){O}
 \tkzDefPoint(5,0){I}
 \tkzDefPoint(0,5){J}
 \tkzInterCC(O,I)(I,O)\tkzGetPoints{B}{C}
 \tkzInterCC(O,I)(J,O)\tkzGetPoints{D}{A}
 \tkzInterCC(I,O)(J,O)\tkzGetPoints{L}{K}
 \tkzDrawArc[angles](O,I)(0,90)
 \tkzDrawArc[angles,color=gray,
             style=dashed](I,O)(90,180)
 \tkzDrawArc[angles,color=gray,
             style=dashed](J,O)(-90,0)
 \tkzDrawPoints(A,B,K)
 \foreach \point in {I,A,B,J,K}{%
             \tkzDrawSegment(O,\point)}
\end{tikzpicture}
```

26.1.9. Option reverse: inversion of the arrow




```
\begin{tikzpicture}
  \tkzDefPoints{0/0/O,3/0/U}
  \tkzDefPoint(10:1){A}
  \tkzDefPoint(90:1){B}
  \tkzLabelPoints(A,B)
  \tkzDrawArc[reverse,tkz arrow={Stealth}](O,A)(B)
  \tkzDrawPoints(A,B,O)
\end{tikzpicture}
```

27. Drawing a sector or sectors

27.1. \tkzDrawSector

Note that the arguments depend on the options.

\tkzDrawSector[⟨local options⟩](⟨O,…⟩)(⟨…⟩)

options	default	definition
towards	towards	O is the center and the arc from A to (OB)
rotate	towards	the arc starts from A and the angle determines its length
R	towards	We give the radius and two angles
R with nodes	towards	We give the radius and two points

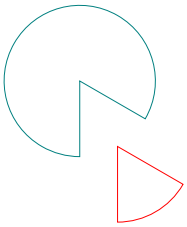
Of course, you can add all the TikZ styles for the drawings.

options	arguments	example
towards	(⟨pt,pt⟩)(⟨pt⟩)	\tkzDrawSector(O,A)(B)
rotate	(⟨pt,pt⟩)(⟨an⟩)	\tkzDrawSector[rotate,color=red](O,A)(90)
R	(⟨pt,r⟩)(⟨an,an⟩)	\tkzDrawSector[R,color=teal](O,2)(30,90)
R with nodes	(⟨pt,r⟩)(⟨pt,pt⟩)	\tkzDrawSector[R with nodes](O,2)(A,B)

Here are a few examples:

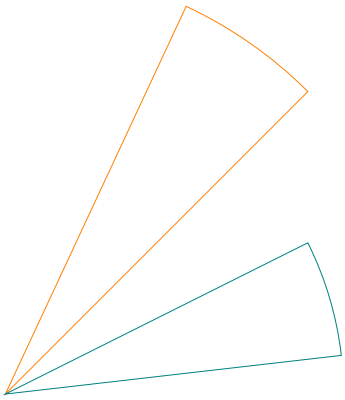
27.1.1. \tkzDrawSector and towards

You don't need to write **towards**. You can use **fill** as an option.



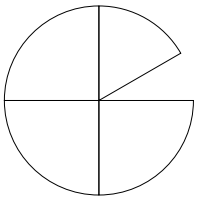
```
\begin{tikzpicture}
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(-30:1){A}
  \tkzDefPointBy[rotation = center O angle -60](A)
  \tkzDrawSector[teal](O,A)(tkzPointResult)
 \begin{scope}[shift={(-60:1)}]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(-30:1){A}
  \tkzDefPointBy[rotation = center O angle -60](A)
  \tkzDrawSector[red](O,tkzPointResult)(A)
  \end{scope}
\end{tikzpicture}
```

27.1.2. \tkzDrawSector and rotate



```
\begin{tikzpicture}[scale=2]
 \tkzDefPoints{0/0/O,2/2/A,2/1/B}
 \tkzDrawSector[rotate,orange](O,A)(20)
 \tkzDrawSector[rotate,teal](O,B)(-20)
\end{tikzpicture}
```

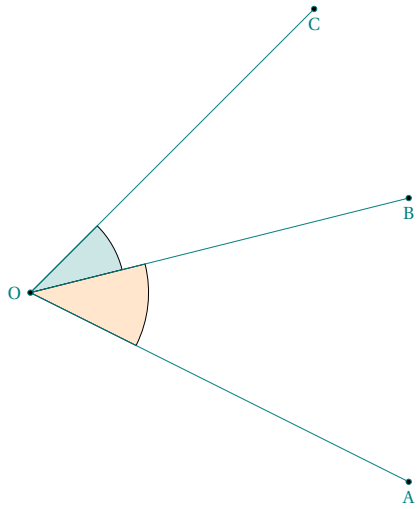
27.1.3. \tkzDrawSector and R



```
\begin{tikzpicture}[scale=1.25]
 \tkzDefPoint(0,0){O}
 \tkzDefPoint(2,-1){A}
 \tkzDrawSector[R](O,1)(30,90)
 \tkzDrawSector[R](O,1)(90,180)
 \tkzDrawSector[R](O,1)(180,270)
 \tkzDrawSector[R](O,1)(270,360)
\end{tikzpicture}
```

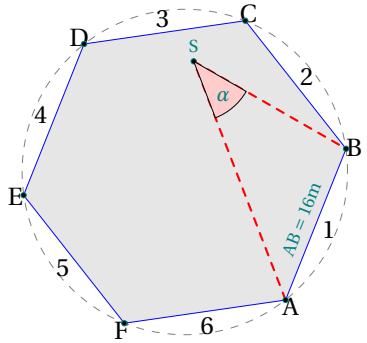
27.1.4. \tkzDrawSector and R with nodes: two sectors

In this example I use the option **fill**, but \tkzFillSector would also work.




```
\begin{tikzpicture}[scale=1.25]
 \tkzDefPoint(0,0){O}
 \tkzDefPoint(4,-2){A}
 \tkzDefPoint(4,1){B}
 \tkzDefPoint(3,3){C}
 \tkzDrawSector[R with nodes,%
                fill=teal!20](O,1)(B,C)
 \tkzDrawSector[R with nodes,%
                fill=orange!20](O,1.25)(A,B)
\tkzDrawSegments(O,A O,B O,C)
\tkzDrawPoints(O,A,B,C)
\tkzLabelPoints(A,B,C)
\tkzLabelPoints[left](O)
\end{tikzpicture}
```

27.1.5. \tkzDrawSector and R with nodes: sector in a hexagon




```
\begin{tikzpicture} [scale=.4]
 \tkzDefPoints{-1/-2/A,1/3/B}
 \tkzDefRegPolygon[side,sides=6](A,B)
 \tkzGetPoint{O}
 \tkzDrawPolygon[fill=black!10, draw=blue](P1,P...,P6)
 \tkzLabelRegPolygon[sep=1.05](O){A,...,F}
 \tkzDrawCircle[dashed](O,A)
 \tkzLabelSegment[above,sloped,
                  midway](A,B){\(A B = 16m\)}
 \foreach \i  [count=\xi from 1]  in {2,...,6,1}
   {%
    \tkzDefMidPoint(P\xi,P\i)
    \path (O) to [pos=1.1] node {\xi} (tkzPointResult) ;
    }
  \tkzDefRandPointOn[segment = P3--P5]
  \tkzGetPoint{S}
  \tkzDrawSegments[thick,dashed,red](A,S S,B)
  \tkzDrawPoints(P1,P...,P6,S)
  \tkzLabelPoint[left,above](S){$S$}
  \tkzDrawSector[R with nodes,fill=red!20](S,2)(A,B)
  \tkzLabelAngle[pos=1.5](A,S,B){$\alpha$}
\end{tikzpicture}
```

28. Filling

28.1. Coloring a disc

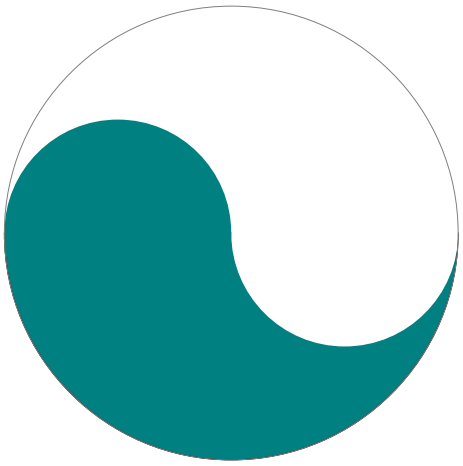
This was possible with the macro \tkzDrawCircle, but the circle had to be drawn; this is no longer the case.

\tkzFillCircle[⟨local options⟩](⟨A,B⟩)

options	default	definition
radius	radius	two points define a radius
R	radius	a point and the measurement of a radius

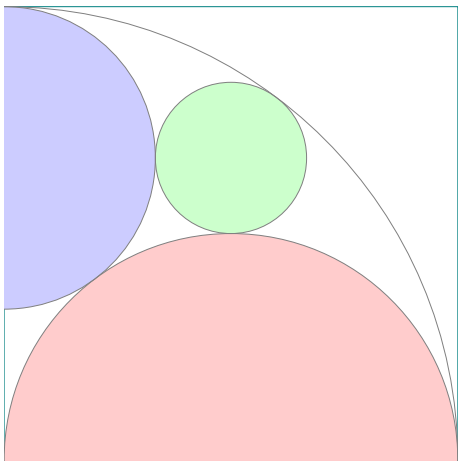
You don't need to write **radius**, because it is the default option. Of course, you can add all the Ti*k*Z styles.

28.1.1. Yin and Yang



```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(-4,0){A}
  \tkzDefPoint(4,0){B}
  \tkzDefPoint(-2,0){I}
  \tkzDefPoint(2,0){J}
  \tkzDrawSector[fill=teal](O,A)(B)
  \tkzFillCircle[fill=white](J,B)
  \tkzFillCircle[fill=teal](I,A)
  \tkzDrawCircle(O,A)
\end{tikzpicture}
```

28.1.2. From a sangaku

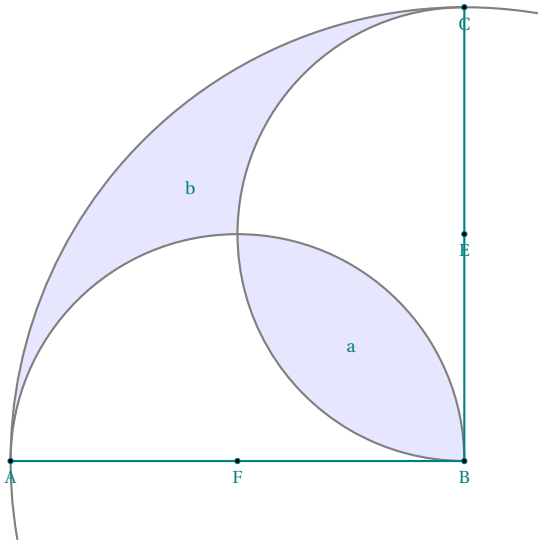


```
\begin{tikzpicture}
   \tkzDefPoint(0,0){B}  \tkzDefPoint(6,0){C}%
   \tkzDefSquare(B,C)    \tkzGetPoints{D}{A}
   \tkzClipPolygon(B,C,D,A)
   \tkzDefMidPoint(A,D)  \tkzGetPoint{F}
   \tkzDefMidPoint(B,C)  \tkzGetPoint{E}
   \tkzDefMidPoint(B,D)  \tkzGetPoint{Q}
   \tkzDefLine[tangent from = B](F,A)
   \tkzGetPoints{H}{G}
   \tkzInterLL(F,G)(C,D) \tkzGetPoint{J}
   \tkzInterLL(A,J)(F,E) \tkzGetPoint{K}
   \tkzDefPointBy[projection=onto B--A](K)
   \tkzGetPoint{M}
   \tkzDrawPolygon(A,B,C,D)
   \tkzFillCircle[red!20](E,B)
   \tkzFillCircle[blue!20](M,A)
   \tkzFillCircle[green!20](K,Q)
  \tkzDrawCircles(B,A M,A E,B K,Q)
\end{tikzpicture}
```

28.1.3. Clipping and filling part I

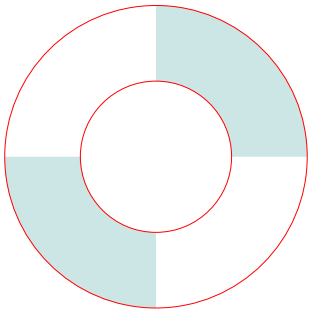

```
\begin{tikzpicture}
\tkzDefPoints{0/0/A,4/0/B,2/2/O,3/4/X,4/1/Y,1/0/Z,
              0/3/W,3/0/R,4/3/S,1/4/T,0/1/U}
\tkzDefSquare(A,B)\tkzGetPoints{C}{D}
\tkzDefPointWith[colinear normed=at X,K=1](O,X)
 \tkzGetPoint{F}
\begin{scope}
  \tkzFillCircle[fill=teal!20](O,F)
  \tkzFillPolygon[white](A,...,D)
  \tkzClipPolygon(A,...,D)
  \foreach \c/\t in {S/C,R/B,U/A,T/D}
  {\tkzFillCircle[teal!20](\c,\t)}
\end{scope}
\foreach \c/\t in {X/C,Y/B,Z/A,W/D}
{\tkzFillCircle[white](\c,\t)}
  \foreach \c/\t in {S/C,R/B,U/A,T/D}
  {\tkzFillCircle[teal!20](\c,\t)}
\end{tikzpicture}
```

28.1.4. Clipping and filling part II




```
\begin{tikzpicture}[scale=.75]
\tkzDefPoints{0/0/A,8/0/B,8/8/C,0/8/D}
\tkzDefMidPoint(A,B) \tkzGetPoint{F}
\tkzDefMidPoint(B,C) \tkzGetPoint{E}
\tkzDefMidPoint(D,B) \tkzGetPoint{I}
\tkzDefMidPoint(I,B) \tkzGetPoint{a}
\tkzInterLC(B,I)(B,C) \tkzGetSecondPoint{K}
\tkzDefMidPoint(I,K) \tkzGetPoint{b}
\begin{scope}
  \tkzFillSector[fill=blue!10](B,C)(A)
  \tkzDefMidPoint(A,B) \tkzGetPoint{x}
  \tkzDrawSemiCircle[fill=white](x,B)
  \tkzDefMidPoint(B,C) \tkzGetPoint{y}
  \tkzDrawSemiCircle[fill=white](y,C)
  \tkzClipCircle(E,B)
  \tkzClipCircle(F,B)
  \tkzFillCircle[fill=blue!10](B,A)
\end{scope}
\tkzDrawSemiCircle[thick](F,B)
\tkzDrawSemiCircle[thick](E,C)
\tkzDrawArc[thick](B,C)(A)
\tkzDrawSegments[thick](A,B B,C)
\tkzDrawPoints(A,B,C,E,F)
\tkzLabelPoints[centered](a,b)
\tkzLabelPoints(A,B,C,E,F)
\end{tikzpicture}
```

28.1.5. Clipping and filling part III



```
\begin{tikzpicture}
  \tkzDefPoint(0,0){A} \tkzDefPoint(1,0){B}
  \tkzDefPoint(2,0){C} \tkzDefPoint(-3,0){a}
  \tkzDefPoint(3,0){b}  \tkzDefPoint(0,3){c}
  \tkzDefPoint(0,-3){d}
\begin{scope}
 \tkzClipPolygon(a,b,c,d)
 \tkzFillCircle[teal!20](A,C)
\end{scope}
 \tkzFillCircle[white](A,B)
 \tkzDrawCircle[color=red](A,C)
 \tkzDrawCircle[color=red](A,B)
\end{tikzpicture}
```

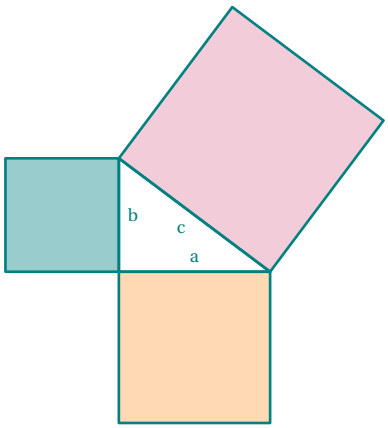
28.2. Coloring a polygon

\tkzFillPolygon[⟨local options⟩](⟨points list⟩)

You can color a polygon while drawing it; this macro colors the inside of the polygon without drawing it.

arguments	example	explanation
(⟨pt1,pt2,…⟩)	(⟨A,B,…⟩)	

28.2.1. \tkzFillPolygon




```
\begin{tikzpicture}[scale=.5]
   \tkzDefPoint(0,0){C} \tkzDefPoint(4,0){A}
   \tkzDefPoint(0,3){B}
   \tkzDefSquare(B,A)       \tkzGetPoints{E}{F}
   \tkzDefSquare(A,C)       \tkzGetPoints{G}{H}
   \tkzDefSquare(C,B)        \tkzGetPoints{I}{J}
   \tkzFillPolygon[color  =  orange!30   ](A,C,G,H)
   \tkzFillPolygon[color  =  teal!40  ](C,B,I,J)
   \tkzFillPolygon[color  =  purple!20](B,A,E,F)
   \tkzDrawPolygon[line width  =  1pt](A,B,C)
   \tkzDrawPolygon[line width  =  1pt](A,C,G,H)
   \tkzDrawPolygon[line width  =  1pt](C,B,I,J)
   \tkzDrawPolygon[line width  =  1pt](B,A,E,F)
   \tkzLabelSegment[above](C,A){$a$}
   \tkzLabelSegment[right](B,C){$b$}
   \tkzLabelSegment[below left](B,A){$c$}
\end{tikzpicture}
```

28.3. \tkzFillSector

Note that the arguments depend on the options.

\tkzFillSector[⟨local options⟩](⟨O,…⟩)(⟨…⟩)

options	default	definition
towards	towards	O is the center and the arc from A to (OB)
rotate	towards	the arc starts from A and the angle determines its length
R	towards	We give the radius and two angles
R with nodes	towards	We give the radius and two points

Of course, you can add all the Ti*k*Z styles for the drawings.

options	arguments	example
towards	(⟨pt,pt⟩)(⟨pt⟩)	\tkzFillSector(O,A)(B)
rotate	(⟨pt,pt⟩)(⟨an⟩)	\tkzFillSector[rotate,color=red](O,A)(90)
R	(⟨pt,r⟩)(⟨an,an⟩)	\tkzFillSector[R,color=blue](O,2)(30,90)
R with nodes	(⟨pt,r⟩)(⟨pt,pt⟩)	\tkzFillSector[R with nodes](O,2)(A,B)

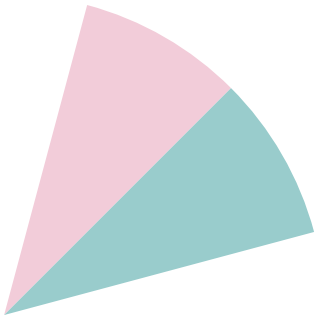
28.3.1. \tkzFillSector and towards

You don't need to write **towards**. Note that the contours are not drawn: only the surface is colored.



```
\begin{tikzpicture}[scale=.6]
\tkzDefPoint(0,0){O}
\tkzDefPoint(-30:3){A}
\tkzDefPointBy[rotation = center O angle -60](A)
\tkzFillSector[fill=purple!20](O,A)(tkzPointResult)
  \begin{scope}[shift={(-60:1)}]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(-30:3){A}
  \tkzDefPointBy[rotation = center O angle -60](A)
  \tkzGetPoint{A'}
  \tkzFillSector[color=teal!40](O,A')(A)
    \end{scope}
\end{tikzpicture}
```

28.3.2. \tkzFillSector and rotate



```
\begin{tikzpicture}[scale=1.5]
 \tkzDefPoint(0,0){O} \tkzDefPoint(2,2){A}
 \tkzFillSector[rotate,color=purple!20](O,A)(30)
 \tkzFillSector[rotate,color=teal!40](O,A)(-30)
\end{tikzpicture}
```

28.4. Coloring an angle: \tkzFillAngle

The simplest case:

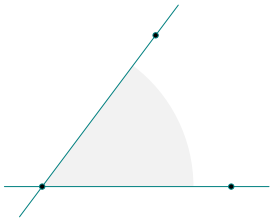
\tkzFillAngle[⟨local options⟩](⟨A,O,B⟩)

O is the vertex of the angle, OA and OB are its sides. Note that the angle depends on the order of the points.

options	default	definition
size	1	this option determines the radius of the colored angular sector.

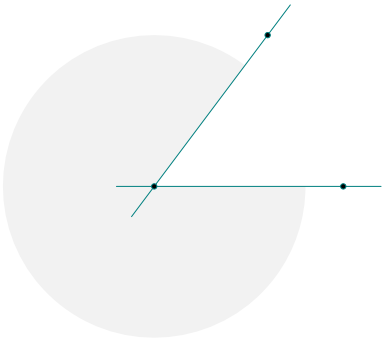
Of course, you can add all the Ti*k*Z styles, such as fill and shade.

28.4.1. Example with size

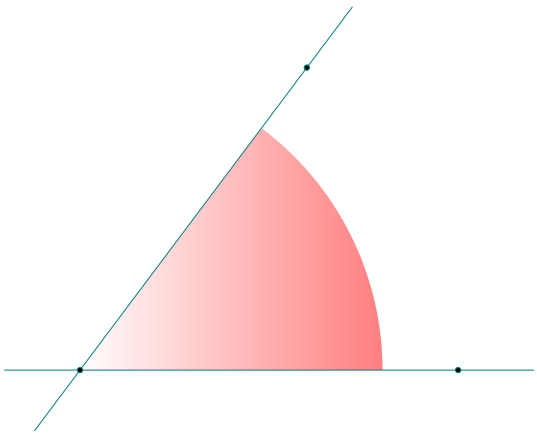


```
\begin{tikzpicture}
   \tkzInit
   \tkzDefPoints{0/0/O,2.5/0/A,1.5/2/B}
   \tkzFillAngle[size=2, fill=gray!10](A,O,B)
   \tkzDrawLines(O,A O,B)
   \tkzDrawPoints(O,A,B)
\end{tikzpicture}
```

28.4.2. Changing the order of items



```
\begin{tikzpicture}
   \tkzInit
   \tkzDefPoints{0/0/O,2.5/0/A,1.5/2/B}
   \tkzFillAngle[size=2,fill=gray!10](B,O,A)
   \tkzDrawLines(O,A O,B)
   \tkzDrawPoints(O,A,B)
\end{tikzpicture}
```

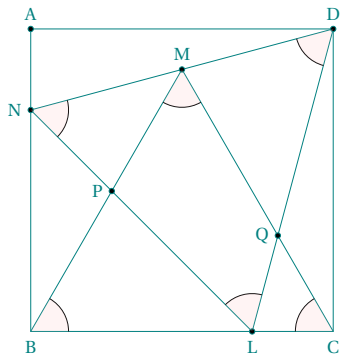


```
\begin{tikzpicture}
   \tkzInit
   \tkzDefPoints{0/0/O,5/0/A,3/4/B}
   % Don't forget {} to get, () to use
   \tkzFillAngle[size=4,left color=white,
                 right color=red!50](A,O,B)
   \tkzDrawLines(O,A O,B)
   \tkzDrawPoints(O,A,B)
\end{tikzpicture}
```

\tkzFillAngles[⟨local options⟩](⟨A,O,B⟩)(⟨A',O',B'⟩)etc.

With common options, there is a macro for several angles.

28.4.3. Several angles




```
\begin{tikzpicture}[scale=0.5]
  \tkzDefPoints{0/0/B,8/0/C,0/8/A,8/8/D}
  \tkzDrawPolygon(B,C,D,A)
  \tkzDefTriangle[equilateral](B,C) \tkzGetPoint{M}
  \tkzInterLL(D,M)(A,B) \tkzGetPoint{N}
  \tkzDefPointBy[rotation=center N angle -60](D)
  \tkzGetPoint{L}
  \tkzInterLL(N,L)(M,B)     \tkzGetPoint{P}
  \tkzInterLL(M,C)(D,L)     \tkzGetPoint{Q}
  \tkzDrawSegments(D,N N,L L,D B,M M,C)
  \tkzDrawPoints(L,N,P,Q,M,A,D)
  \tkzLabelPoints[left](N,P,Q)
  \tkzLabelPoints[above](M,A,D)
  \tkzLabelPoints(L,B,C)
  \tkzMarkAngles(C,B,M B,M,C M,C,B D,L,N L,N,D N,D,L)
  \tkzFillAngles[fill=red!20,opacity=.2](C,B,M
      B,M,C M,C,B D,L,N L,N,D N,D,L)
\end{tikzpicture}
```

29. Controlling the bounding box

From the PgfManual:
"When you add the clip option, the current path is used for clipping subsequent drawings. Clipping never enlarges the clipping area. Thus, when you clip against a certain path and then clip again against another path, you clip against the intersection of both. The only way to enlarge the clipping path is to end the pgfscope in which the clipping was done. At the end of a pgfscope the clipping path that was in force at the beginning of the scope is reinstalled."
First of all, with Ti*k*Z you don't have to deal with the size of the bounding box. Early versions of tkz-euclide did not control it; since version 4, the size of the bounding box is limited.
The initial bounding box after using the macro **\tkzInit** is defined by the rectangle based on the points $(0,0)$ and $(10,10)$. The macro **\tkzInit** modifies this initial bounding box with the arguments **xmin**, **xmax**, **ymin** and **ymax**. Of course, anything drawn outside modifies the bounding box, which Ti*k*Z maintains. You can influence this behavior with Ti*k*Z commands or options, such as **\useasboundingbox** or the option **use as bounding box**. You can then reserve a box for a figure, but the figure may overflow the box and spread over the main text. The command **\pgfresetboundingbox** clears the bounding box and starts a new one.

29.1. Utility of \tkzInit

However, it is sometimes necessary to control the size of what is displayed. To do this, you need to prepare the bounding box you are going to work in: this is the role of the macro **\tkzInit**. For some drawings, it is useful to fix the extreme values (xmin, xmax, ymin and ymax) and to clip this rectangle, in order to control the size of the figure as well as possible.
Two macros are useful to control the bounding box:

- **\tkzInit**
- **\tkzClip**

I have also added macros directly linked to the bounding box: you can display it or clip it (see below).

29.2. \tkzInit

\tkzInit[⟨local options⟩]

options	default	definition
xmin	0	minimum value of the abscissae in cm
xmax	10	maximum value of the abscissae in cm
xstep	1	difference between two graduations in x
ymin	0	minimum y-axis value in cm
ymax	10	maximum y-axis value in cm
ystep	1	difference between two graduations in y

The role of **\tkzInit** is to define an orthogonal coordinate system and a rectangular part of the plane, in which you place your drawings with Cartesian coordinates. This macro defines your working area, as on a graphing calculator. Since version 4 of **tkz-euclide**, **\xstep** and **\ystep** are always 1, so **\tkzInit** is mostly useful for actions such as clipping out.

29.3. \tkzClip

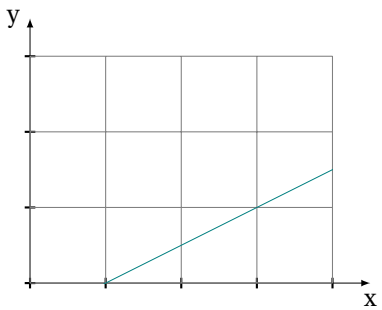
\tkzClip[⟨local options⟩]

This macro hides everything outside the rectangle defined by (xmin ; ymin) and (xmax ; ymax).

options	default	definition
space	1	added value on the right, left, bottom and top of the background

The option **space** enlarges the visible part of the drawing, which becomes the rectangle defined by (xmin-space ; ymin-space) and (xmax+space ; ymax+space). **space** can be negative! The unit is the cm and must not be written.

This macro clips the initial rectangle, so that only the paths inside this rectangle are drawn.



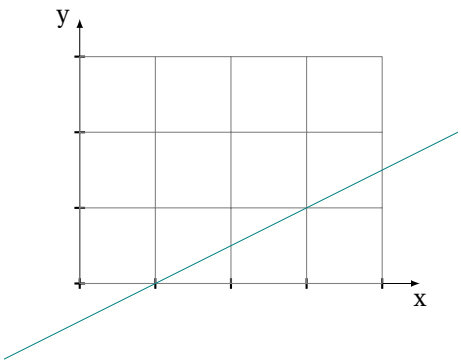

```
\begin{tikzpicture}
 \tkzInit[xmax=4, ymax=3]
 \tkzDefPoints{-1/-1/A,5/2/B}
 \tkzDrawX \tkzDrawY
 \tkzGrid
 \tkzClip
 \tkzDrawSegment(A,B)
\end{tikzpicture}
```

You can add a bit of space:

```
\tkzClip[space=1]
```

29.4. \tkzClip and the option space

This option adds some space around the clipped rectangle.




```
\begin{tikzpicture}
 \tkzInit[xmax=4, ymax=3]
 \tkzDefPoints{-1/-1/A,5/2/B}
 \tkzDrawX \tkzDrawY
 \tkzGrid
 \tkzClip[space=1]
 \tkzDrawSegment(A,B)
\end{tikzpicture}
```

The dimensions of the clipped rectangle are **xmin-1**, **ymin-1**, **xmax+1** and **ymax+1**.

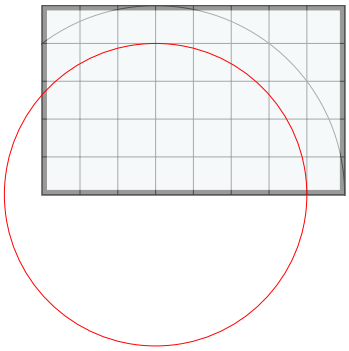
29.5. \tkzShowBB

The simplest macro.

\tkzShowBB[⟨local options⟩]

This macro displays the bounding box as a rectangular frame. It accepts Ti*k*Z options.

29.5.1. Example with \tkzShowBB



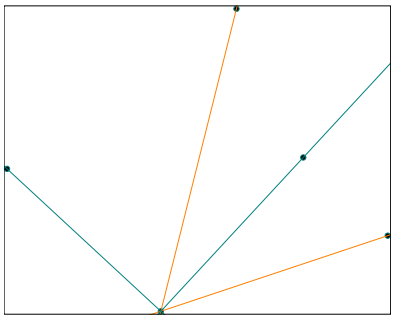
```
\begin{tikzpicture}[scale=.5]
  \tkzInit[ymax=5,xmax=8]
  \tkzGrid
  \tkzDefPoint(3,0){A}
   \begin{scope}
    \tkzClipBB
    \tkzDefCircle[R](A,5) \tkzGetPoint{a}
    \tkzDrawCircle(A,a)
    \tkzShowBB[line width = 4pt,fill=teal!10,%
               opacity=.4]
   \end{scope}
\tkzDefCircle[R](A,4) \tkzGetPoint{b}
\tkzDrawCircle[red](A,b)
\end{tikzpicture}
```

29.6. \tkzClipBB

\tkzClipBB

The idea is to limit the next constructions to the current bounding box.

29.6.1. Example with \tkzClipBB and the bisectors



```
\begin{tikzpicture}
\tkzInit[xmin=-3,xmax=6, ymin=-1,ymax=6]
\tkzDefPoint(0,0){O}\tkzDefPoint(3,1){I}
\tkzDefPoint(1,4){J}
\tkzDefLine[bisector](I,O,J) \tkzGetPoint{i}
\tkzDefLine[bisector out](I,O,J) \tkzGetPoint{j}
\tkzDrawPoints(O,I,J,i,j)
\tkzClipBB
\tkzDrawLines[add = 1 and 2,color=orange](O,I O,J)
\tkzDrawLines[add = 1 and 2](O,i O,j)
\tkzShowBB
\end{tikzpicture}
```

30. Clipping different objects

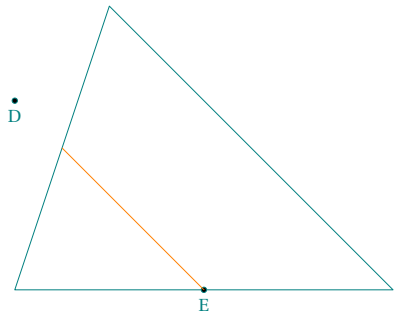
30.1. Clipping a polygon

\tkzClipPolygon[⟨local options⟩](⟨points list⟩)

This macro restricts the drawings to the given polygon.

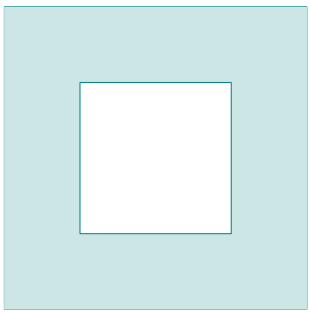
arguments	example	explanation
(⟨pt1,pt2,pt3,…⟩)	(⟨A,B,C⟩)	
options	default	definition
out		clips the outside of the object

30.1.1. \tkzClipPolygon



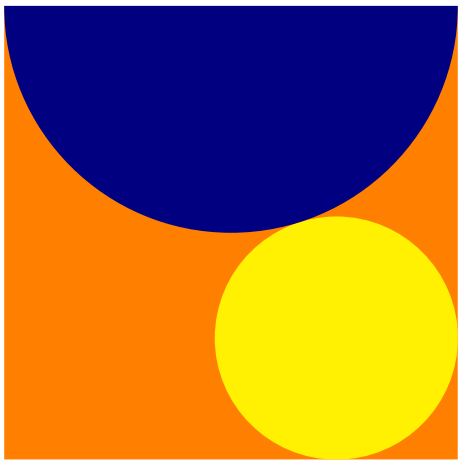

```
\begin{tikzpicture}[scale=1.25]
\tkzDefPoint(0,0){A}
\tkzDefPoint(4,0){B}
\tkzDefPoint(1,3){C}
\tkzDrawPolygon(A,B,C)
\tkzDefPoint(0,2){D}
\tkzDefPoint(2,0){E}
\tkzDrawPoints(D,E)
\tkzLabelPoints(D,E)
\tkzClipPolygon(A,B,C)
\tkzDrawLine[new](D,E)
\end{tikzpicture}
```

30.1.2. \tkzClipPolygon[out]



```
\begin{tikzpicture}[scale=1]
\tkzDefPoint(0,0){P1}
\tkzDefPoint(4,0){P2}
\tkzDefPoint(4,4){P3}
\tkzDefPoint(0,4){P4}
\tkzDefPoint(1,1){Q1}
\tkzDefPoint(3,1){Q2}
\tkzDefPoint(3,3){Q3}
\tkzDefPoint(1,3){Q4}
\tkzDrawPolygon(P1,P2,P3,P4)
\begin{scope}
   \tkzClipPolygon[out](Q1,Q2,Q3,Q4)
  \tkzFillPolygon[teal!20](P1,P2,P3,P4)
\end{scope}
\tkzDrawPolygon(Q1,Q2,Q3,Q4)
\end{tikzpicture}
```

30.1.3. Example: use of Clip for Sangaku in a square



```
\begin{tikzpicture}[scale=.75]
 \tkzDefPoint(0,0){A} \tkzDefPoint(8,0){B}
 \tkzDefSquare(A,B)   \tkzGetPoints{C}{D}
 \tkzDefPoint(4,8){F}
 \tkzDefTriangle[equilateral](C,D)
 \tkzGetPoint{I}
 \tkzDefPointBy[projection=onto B--C](I)
 \tkzGetPoint{J}
 \tkzInterLL(D,B)(I,J)  \tkzGetPoint{K}
 \tkzDefPointBy[symmetry=center K](B)
 \tkzGetPoint{M}
 \tkzClipPolygon(B,C,D,A)
 \tkzFillPolygon[color = orange](A,B,C,D)
 \tkzFillCircle[color = yellow](M,I)
 \tkzFillCircle[color = blue!50!black](F,D)
\end{tikzpicture}
```

30.2. Clipping a disc

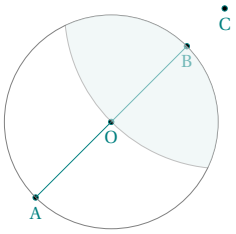
`\tkzClipCircle[⟨local options⟩](⟨A,B⟩)`

arguments	example	explanation
(⟨A,B⟩)	(⟨A,B⟩)	AB radius

options	default	definition
out		clips the outside of the object

You don't need to write **radius**, because it is the default option.

30.2.1. Simple clip



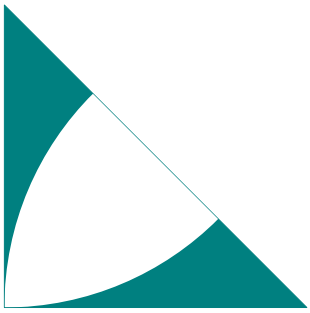

```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoint(0,0){A} \tkzDefPoint(2,2){O}
  \tkzDefPoint(4,4){B} \tkzDefPoint(5,5){C}
  \tkzDrawPoints(O,A,B,C)
  \tkzLabelPoints(O,A,B,C)
  \tkzDrawCircle(O,A)
  \tkzClipCircle(O,A)
  \tkzDrawLine(A,C)
  \tkzDrawCircle[fill=teal!10,opacity=.5](C,O)
\end{tikzpicture}
```

30.3. Clip out



```
\begin{tikzpicture}
  \tkzInit[xmin=-3,ymin=-2,xmax=4,ymax=3]
   \tkzDefPoint(0,0){O}
   \tkzDefPoint(-4,-2){A}
   \tkzDefPoint(3,1){B}
   \tkzDefCircle[R](O,2) \tkzGetPoint{o}
   \tkzDrawPoints(A,B) % to have a good bounding box
   \begin{scope}
    \tkzClipCircle[out](O,o)
    \tkzDrawLines(A,B)
   \end{scope}
\end{tikzpicture}
```

30.4. Intersection of disks



```
\begin{tikzpicture}
\tkzDefPoints{0/0/O,4/0/A,0/4/B}
\tkzDrawPolygon[fill=teal](O,A,B)
\tkzClipPolygon(O,A,B)
\tkzClipCircle(A,O)
\tkzClipCircle(B,O)
\tkzFillPolygon[white](O,A,B)
\end{tikzpicture}
```

See a more complex example of clipping in section 51.6.

30.5. Clipping a sector

Note that the arguments depend on the options.

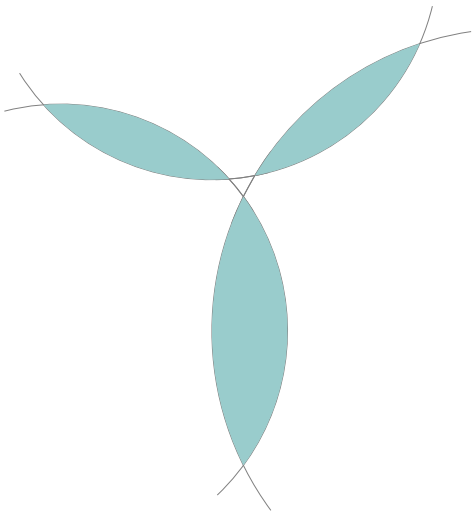
`\tkzClipSector[⟨local options⟩](⟨O,…⟩)(⟨…⟩)`

options	default	definition
towards	towards	O is the center and the sector starts from A to (OB)
rotate	towards	The sector starts from A and the angle determines its amplitude.
R	towards	We give the radius and two angles

Of course, you can add all the Ti*k*Z styles for the drawings.

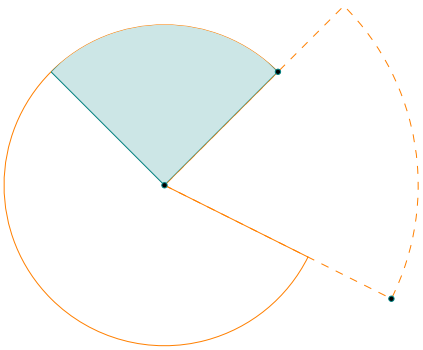
options	arguments	example
towards	(⟨pt,pt⟩)(⟨pt⟩)	`\tkzClipSector(O,A)(B)`
rotate	(⟨pt,pt⟩)(⟨angle⟩)	`\tkzClipSector[rotate](O,A)(90)`
R	(⟨pt,r⟩)(⟨angle 1,angle 2⟩)	`\tkzClipSector[R](O,2)(30,90)`

30.5.1. Clipping a sector: first example



```
\begin{tikzpicture}[scale=0.5]
\tkzDefPoint(0,0){a}
\tkzDefPoint(12,0){b}
\tkzDefPoint(4,10){c}
\tkzInterCC[R](a,6)(b,8)
\tkzGetFirstPoint{AB1} \tkzGetSecondPoint{AB2}
\tkzInterCC[R](a,6)(c,6)
\tkzGetFirstPoint{AC1} \tkzGetSecondPoint{AC2}
\tkzInterCC[R](b,8)(c,6)
\tkzGetFirstPoint{BC1} \tkzGetSecondPoint{BC2}
\tkzDrawArc(a,AB2)(AB1)
\tkzDrawArc(b,AB1)(AB2)
\tkzDrawArc(a,AC2)(AC1)
\tkzDrawArc(c,AC1)(AC2)
\tkzDrawArc(b,BC2)(BC1)
\tkzDrawArc(c,BC1)(BC2)
\begin{scope}
\tkzClipSector(b,BC2)(BC1)
\tkzFillSector[teal!40!white](c,BC1)(BC2)
\end{scope}
\begin{scope}
\tkzClipSector(a,AB2)(AB1)
\tkzFillSector[teal!40!white](b,AB1)(AB2)
\end{scope}
\begin{scope}
\tkzClipSector(a,AC2)(AC1)
\tkzFillSector[teal!40!white](c,AC1)(AC2)
\end{scope}
\end{tikzpicture}
```

30.5.2. Clipping a sector: second example



```
\begin{tikzpicture}[scale=1.5]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2,-1){A}
  \tkzDefPoint(1,1){B}
  \tkzDrawSector[new,dashed](O,A)(B)
  \tkzDrawSector[new](O,B)(A)
\begin{scope}
\tkzClipSector(O,B)(A)
\tkzDefSquare(O,B) \tkzGetPoints{B'}{O'}
\tkzDrawPolygon[color=teal,fill=teal!20](O,B,B',O')
\end{scope}
\tkzDrawPoints(A,B,O)
\end{tikzpicture}
```

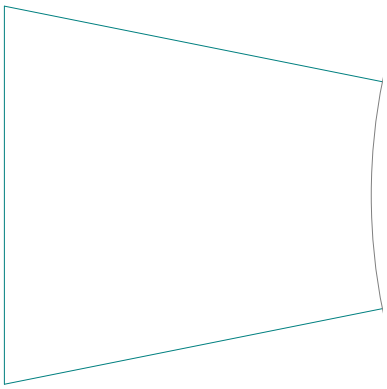
30.6. TikZ options: trim left or trim right

See the pgfmanual.

30.7. TikZ Controls \pgfinterruptboundingbox and \endpgfinterruptboundingbox

This command temporarily interrupts the calculation of the bounding box and starts a new one. See the pgfmanual

30.7.1. Controlling the bounding box



```
\begin{tikzpicture}
\tkzDefPoint(0,5){A}\tkzDefPoint(5,4){B}
\tkzDefPoint(0,0){C}\tkzDefPoint(5,1){D}
\tkzDrawSegments(A,B C,D A,C)
\pgfinterruptboundingbox
   \tkzInterLL(A,B)(C,D)\tkzGetPoint{I}
\endpgfinterruptboundingbox
\tkzClipBB
\tkzDrawCircle(I,B)
\end{tikzpicture}
```

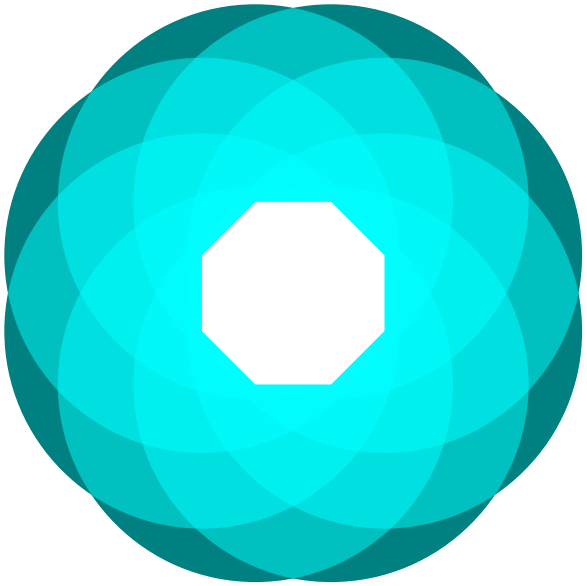
30.8. Reverse clip: tkzreverseclip

To use this option, a bounding box must be defined.

```
\tikzset{tkzreverseclip/.style={insert path={
   (current bounding box.south west) --(current bounding box.north west)
 --(current bounding box.north east) --  (current bounding box.south east)
 -- cycle} }}
```

30.8.1. Example with \tkzClipPolygon[out]

\tkzClipPolygon[out], \tkzClipCircle[out] use this option.



```
\begin{tikzpicture}[scale=1]
\tkzInit[xmin=-5,xmax=5,ymin=-4,ymax=6]
\tkzClip
\tkzDefPoints{-.5/0/P1,.5/0/P2}
\foreach \i [count=\j from 3] in {2,...,7}{%
    \tkzDefShiftPoint[P\i]({45*(\i-1)}:1){P\j}}
\tkzClipPolygon[out](P1,P...,P8)
\tkzCalcLength(P1,P5)\tkzGetLength{r}
\begin{scope}[blend group=screen]
  \foreach \i in {1,...,8}{%
   \tkzDefCircle[R](P\i,\r) \tkzGetPoint{x}
   \tkzFillCircle[color=teal](P\i,x)}
  \end{scope}
\end{tikzpicture}
```

Part VII.

Marking and labelling

31. Marking

31.1. Mark a segment \tkzMarkSegment

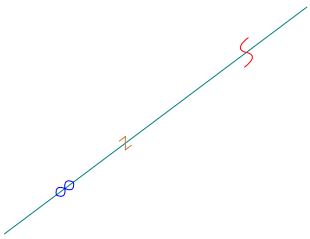
\tkzMarkSegment[⟨local options⟩](⟨pt1,pt2⟩)

This macro places a mark on a segment.

options	default	definition
pos	.5	position of the mark
color	black	color of the mark
mark	none	choice of the mark
size	4pt	size of the mark

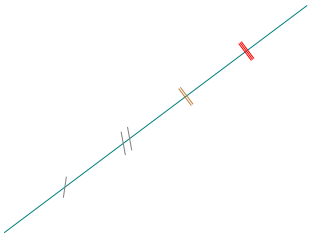
Possible marks are those provided by Ti*k*Z, but other marks have been created based on an idea by Yves Combe.

31.1.1. Several marks



```
\begin{tikzpicture}
  \tkzDefPoint(2,1){A}
  \tkzDefPoint(6,4){B}
  \tkzDrawSegment(A,B)
  \tkzMarkSegment[color=brown,size=2pt,pos=0.4, mark=z](A,B)
  \tkzMarkSegment[color=blue,pos=0.2, mark=oo](A,B)
  \tkzMarkSegment[pos=0.8,mark=s,color=red](A,B)
\end{tikzpicture}
```

31.1.2. Use of mark



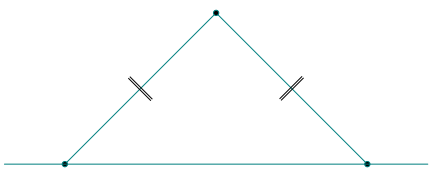
```
\begin{tikzpicture}
  \tkzDefPoint(2,1){A}
  \tkzDefPoint(6,4){B}
  \tkzDrawSegment(A,B)
  \tkzMarkSegment[color=gray,pos=0.2,mark=s|](A,B)
  \tkzMarkSegment[color=gray,pos=0.4,mark=s||](A,B)
  \tkzMarkSegment[color=brown,pos=0.6,mark=||](A,B)
  \tkzMarkSegment[color=red,pos=0.8,mark=|||](A,B)
\end{tikzpicture}
```

31.2. Marking segments \tkzMarkSegments

\tkzMarkSegments[⟨local options⟩](⟨pt1,pt2 pt3,pt4 ...⟩)

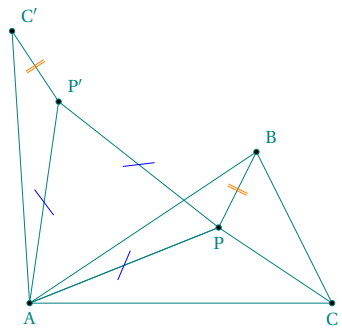
The arguments are a list of pairs of points separated by spaces. The Ti*k*Z styles are available.

31.2.1. Marks for an isosceles triangle



```
\begin{tikzpicture}[scale=1]
 \tkzDefPoints{0/0/O,2/2/A,4/0/B,6/2/C}
 \tkzDrawSegments(O,A A,B)
 \tkzDrawPoints(O,A,B)
 \tkzDrawLine(O,B)
 \tkzMarkSegments[mark=||,size=6pt](O,A A,B)
\end{tikzpicture}
```

31.3. Marking with a construction: equilateral triangle




```
\begin{tikzpicture}[scale=1]
 \tkzDefPoint(0,0){A}\tkzDefPoint(3,2){B}
 \tkzDefPoint(4,0){C}\tkzDefPoint(2.5,1){P}
 \tkzDrawPolygon(A,B,C)
 \tkzDefEquilateral(A,P) \tkzGetPoint{P'}
 \tkzDefPointsBy[rotation=center A angle 60](P,B){P',C'}
 \tkzDrawPolygon(A,P,P')
 \tkzDrawPolySeg(P',C',A,P,B)
 \tkzDrawSegment(C,P)
 \tkzDrawPoints(A,B,C,C',P,P')
 \tkzMarkSegments[mark=s|,size=6pt,
 color=blue](A,P P,P' P',A)
 \tkzMarkSegments[mark=||,color=orange](B,P P',C')
 \tkzLabelPoints(A,C) \tkzLabelPoints[below](P)
 \tkzLabelPoints[above right](P',C',B)
\end{tikzpicture}
```

31.4. Mark an arc \tkzMarkArc

`\tkzMarkArc[⟨local options⟩](⟨pt1,pt2,pt3⟩)`

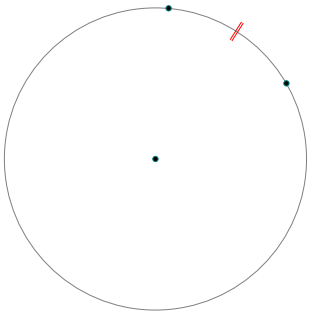
This macro places a mark on an arc: pt1 is the center, pt2 and pt3 are the endpoints of the arc.

options	default	definition
pos	.5	position of the mark
color	black	color of the mark
mark	none	choice of the mark
size	4pt	size of the mark

Possible marks are those provided by Ti*k*Z, but other marks have been created based on an idea by Yves Combe.

```
|, ||,|||, z, s, x, o, oo
```

31.4.1. Several marks



```
\begin{tikzpicture}
\tkzDefPoint(0,0){O}
\pgfmathsetmacro\r{2}
\tkzDefPoint(30:\r){A}
\tkzDefPoint(85:\r){B}
\tkzDrawCircle(O,A)
\tkzMarkArc[color=red,mark=||](O,A,B)
\tkzDrawPoints(B,A,O)
\end{tikzpicture}
```

31.5. Marking an angle: \tkzMarkAngle

More delicate operation because there are many options. The symbols used for marking in addition to those of Ti*k*Z are defined in the file `tkz-lib-marks.tex` and designated by the following characters:

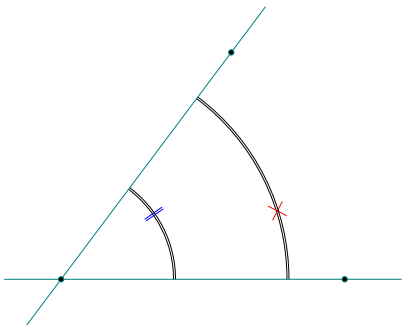
|, ||,|||, z, s, x, o, oo

\tkzMarkAngle[⟨local options⟩](⟨A,O,B⟩)

O is the vertex. Several markings are possible: you can simply draw an arc, or add a mark on this arc. The style of the arc is chosen with the option **arc**, its radius is given by **size** and the size of the mark by **mksize**; the arc can, of course, be colored.

options	default	definition
arc	l	choice of l, ll and lll (single, double or triple).
size	1 (cm)	arc radius.
mark	none	choice of mark.
mksize	4pt	symbol size (mark).
mkcolor	black	symbol color (mark).
mkpos	0.5	position of the symbol on the arc.

31.5.1. Example with mark = x and with mark =||



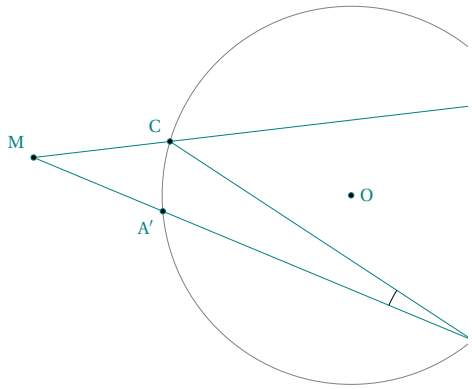
```
\begin{tikzpicture}[scale=.75]
    \tkzDefPoints{0/0/O,5/0/A,3/4/B}
    \tkzMarkAngle[size = 4,mark = x,
                  arc=ll,mkcolor = red,mkpos=.33](A,O,B)
    \tkzMarkAngle[size = 2,mark = ||,
                arc=ll,mkcolor = blue,mkpos=.66](A,O,B)
    \tkzDrawLines(O,A O,B)
    \tkzDrawPoints(O,A,B)
\end{tikzpicture}
```

\tkzMarkAngles[⟨local options⟩](⟨A,O,B⟩)(⟨A',O',B'⟩)etc.

With common options, there is a macro for several angles.

31.6. Marking a small angle: option veclen

The problem comes from the "decorate" action and from the value of **size** in **\tkzMarkAngle**. The solution is to place **\tkzMarkAngle** in a scope with the option **veclen**. In the next example, without the scope the result is: LaTeX Error: Dimension too large.




```
\begin{tikzpicture}[scale=1]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2.5,0){N}
  \tkzDefPoint(-4.2,0.5){M}
  \tkzDefPointBy[rotation=center O angle 30](N)
  \tkzGetPoint{B}
  \tkzDefPointBy[rotation=center O angle -50](N)
  \tkzGetPoint{A}
  \tkzInterLC[common=B](M,B)(O,B) \tkzGetFirstPoint{C}
  \tkzInterLC[common=A](M,A)(O,A) \tkzGetFirstPoint{A'}
  \tkzDrawSegments(A,C M,A M,B A,B)
  \tkzDrawCircle(O,N)
  \begin{scope}[veclen]
    \tkzMarkAngle[mkpos=.2, size=1.2](C,A,M)
  \end{scope}
  \tkzDrawPoints(O, A, B, M, B, C, A')
  \tkzLabelPoints[right](O,A,B)
  \tkzLabelPoints[above left](M,C)
  \tkzLabelPoint[below left](A'){$A'$}
\end{tikzpicture}
```

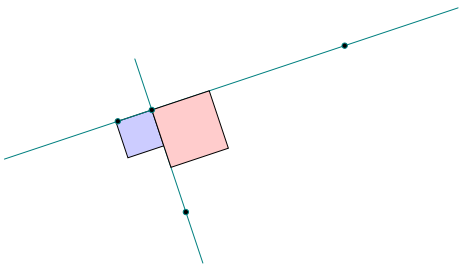
31.7. Marking a right angle: \tkzMarkRightAngle

\tkzMarkRightAngle[⟨local options⟩](⟨A,O,B⟩)

The option **german** changes the style of the mark, and the option **size** its size.

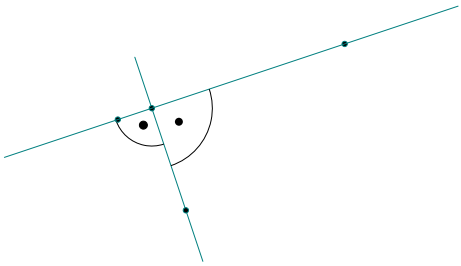
options	default	definition
german	normal	german arc with inner point.
size	0.2	side size.

31.7.1. Marking a right angle



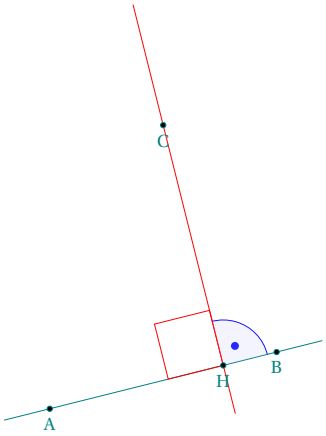
```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,3/1/B,0.9/-1.2/P}
  \tkzDefPointBy[projection = onto B--A](P)  \tkzGetPoint{H}
  \tkzDrawLines[add=.5 and .5](P,H)
  \tkzMarkRightAngle[fill=blue!20,size=.5,draw](A,H,P)
  \tkzDrawLines[add=.5 and .5](A,B)
  \tkzMarkRightAngle[fill=red!20,size=.8](B,H,P)
  \tkzDrawPoints[](A,B,P,H)
\end{tikzpicture}
```

31.7.2. Marking a right angle, German style



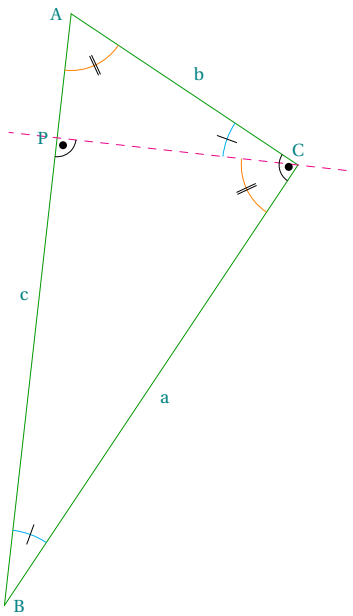
```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,3/1/B,0.9/-1.2/P}
  \tkzDefPointBy[projection = onto B--A](P)  \tkzGetPoint{H}
  \tkzDrawLines[add=.5 and .5](P,H)
  \tkzMarkRightAngle[german,size=.5,draw](A,H,P)
  \tkzDrawPoints[](A,B,P,H)
  \tkzDrawLines[add=.5 and .5](A,B)
  \tkzMarkRightAngle[german,size=.8](P,H,B)
\end{tikzpicture}
```

31.7.3. Mix of styles




```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(4,1){B}
  \tkzDefPoint(2,5){C}
  \tkzDefPointBy[projection=onto B--A](C)
      \tkzGetPoint{H}
  \tkzDrawLine(A,B)
  \tkzDrawLine[add = .5 and .2,color=red](C,H)
  \tkzMarkRightAngle[,size=1,color=red](C,H,A)
  \tkzMarkRightAngle[german,size=.8,color=blue](B,H,C)
  \tkzFillAngle[opacity=.2,fill=blue!20,size=.8](B,H,C)
  \tkzLabelPoints(A,B,C,H)
  \tkzDrawPoints(A,B,C,H)
\end{tikzpicture}
```

31.7.4. Full example




```
\begin{tikzpicture}[rotate=-90]
\tkzDefPoint(0,1){A}
\tkzDefPoint(2,4){C}
\tkzDefPointWith[orthogonal normed,K=7](C,A)
\tkzGetPoint{B}
\tkzDrawSegment[green!60!black](A,C)
\tkzDrawSegment[green!60!black](C,B)
\tkzDrawSegment[green!60!black](B,A)
\tkzDefSpcTriangle[orthic](A,B,C){N,O,P}
\tkzDrawLine[dashed,color=magenta](C,P)
\tkzLabelPoint[left](A){$A$}
\tkzLabelPoint[right](B){$B$}
\tkzLabelPoint[above](C){$C$}
\tkzLabelPoint[left](P){$P$}
\tkzLabelSegment[auto](B,A){$c$}
\tkzLabelSegment[auto,swap](B,C){$a$}
\tkzLabelSegment[auto,swap](C,A){$b$}
\tkzMarkAngle[size=1,color=cyan,mark=|](C,B,A)
\tkzMarkAngle[size=1,color=cyan,mark=|](A,C,P)
\tkzMarkAngle[size=0.75,color=orange,
    mark=||](P,C,B)
\tkzMarkAngle[size=0.75,color=orange,
   mark=||](B,A,C)
\tkzMarkRightAngle[german](A,C,B)
\tkzMarkRightAngle[german](B,P,C)
\end{tikzpicture}
```

31.8. \tkzMarkRightAngles

`\tkzMarkRightAngles[⟨local options⟩](⟨A,O,B⟩)(⟨A',O',B'⟩)etc.`

With common options, there is a macro for several angles.

31.9. The TikZ library angles

If you prefer the Ti*k*Z library **angles**, you can mark angles with the macros `\tkzPicAngle` and `\tkzPicRightAngle`.

`\tkzPicAngle[⟨tikz options⟩](⟨A,O,B⟩)`

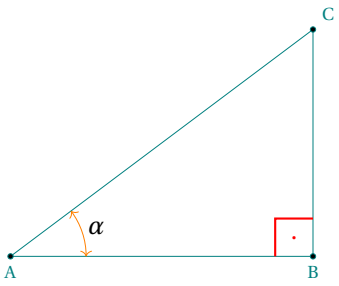
options	example	definition
tikz option	see below	drawing of the angle $\widehat{AOB}$.

`\tkzPicRightAngle[⟨tikz options⟩](⟨A,O,B⟩)`

options	example	definition
tikz option	see below	drawing of the right angle $\widehat{AOB}$.

You need to know the options of the library **angles**.

31.9.1. Angle with TikZ




```
\begin{tikzpicture}
\tkzDefPoints{0/0/A,4/0/B}
\tkzDefTriangle[right,swap](A,B) \tkzGetPoint{C}
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints(A,B,C)
\tkzLabelPoints[below](B,A)
\tkzLabelPoints[above right](C)
\tkzPicAngle["$\alpha$",draw=orange,
            <->,angle eccentricity=1.2,
            angle radius=1cm](B,A,C)
\tkzPicRightAngle[draw,red,thick,
             angle eccentricity=.5,
             pic text=.](C,B,A)
\end{tikzpicture}
```

32. Labelling points

32.1. Label for a point: \tkzLabelPoint

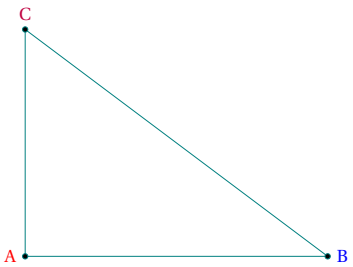
You can add several labels to the same point by using this macro several times.

\tkzLabelPoint[⟨local options⟩](⟨point⟩){⟨label⟩}

arguments	example	
point	\tkzLabelPoint(A){A_1}	
options	default	definition
TikZ options		color, position etc.

Optionally, you can use any TikZ style, in particular placements such as above, right, …

32.1.1. Example with \tkzLabelPoint




```
\begin{tikzpicture}
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(4,0){B}
  \tkzDefPoint(0,3){C}
  \tkzDrawSegments(A,B B,C C,A)
  \tkzDrawPoints(A,B,C)
  \tkzLabelPoint[left,red](A){$A$}
  \tkzLabelPoint[right,blue](B){$B$}
  \tkzLabelPoint[above,purple](C){$C$}
\end{tikzpicture}
```

32.1.2. Label and reference

The reference of a point is what lets you use the point in the code; the label is the name of the point that is displayed.




```
\begin{tikzpicture}
   \tkzDefPoint(2,0){A}
   \tkzDrawPoint(A)
   \tkzLabelPoint[above](A){$A_1$}
 \end{tikzpicture}
```

32.2. Add labels to points \tkzLabelPoints

You can quickly place several labels when the references of the points are identical to the labels and when all the labels are placed in the same way relative to their points. By default, **below right** is used.

\tkzLabelPoints[⟨local options⟩](⟨A1,A2,...⟩)

arguments	example	result
list of points	\tkzLabelPoints(A,B,C)	Display of A, B and C

This macro reduces the number of lines of code, but all the points rarely need the same label position.

32.2.1. Example with \tkzLabelPoints




```
\begin{tikzpicture}
  \tkzDefPoint(2,3){A}
  \tkzDefShiftPoint[A](30:2){B}
  \tkzDefShiftPoint[A](30:5){C}
  \tkzDrawPoints(A,B,C)
  \tkzLabelPoints(A,B,C)
\end{tikzpicture}
```

32.3. Automatic position of labels \tkzAutoLabelPoints

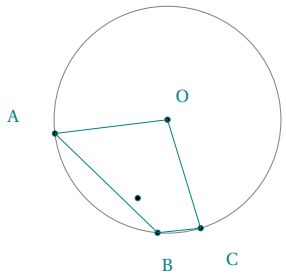
The label of a point is placed in the direction defined by a center, given with the option **center**, and the point. The distance between the label and the point is a percentage of the distance between the center and the point, given by **dist**.

\tkzAutoLabelPoints[⟨local options⟩](⟨A1,A2,...⟩)

arguments	example	result
list of points	\tkzAutoLabelPoint(A,B,C)	Display of A, B and C

32.3.1. Label for points with \tkzAutoLabelPoints

Here the labels are positioned relative to the centroid of A, B, C and O.




```
\begin{tikzpicture}[scale=1]
 \tkzDefPoint(2,1){O}
 \tkzDefRandPointOn[circle=center O radius 1.5]\tkzGetPoint{A}
 \tkzDefPointBy[rotation=center O angle 100](A)\tkzGetPoint{C}
 \tkzDefPointBy[rotation=center O angle 78](A)\tkzGetPoint{B}
 \tkzDrawCircle(O,A)
 \tkzDrawPoints(O,A,B,C)
 \tkzDrawSegments(C,B B,A A,O O,C)
 \tkzDefTriangleCenter[centroid](A,B,C) \tkzGetPoint{G}
 \tkzDrawPoint(tkzPointResult)
 \tkzAutoLabelPoints[center=G,dist=.5](O,A,B,C)
\end{tikzpicture}
```

33. Labelling segments

33.1. The macros \tkzLabelSegment and \tkzLabelSegments

\tkzLabelSegment[⟨local options⟩](⟨pt1,pt2⟩){⟨label⟩}

This macro places a label along a segment or a line. The options are those of Ti*k*Z, for example **pos**.

argument	example	definition
label	\tkzLabelSegment(A,B){5}	label text
(pt1,pt2)	(A,B)	label along [AB]

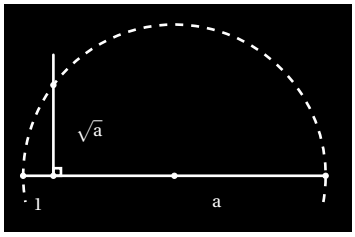
options	default	definition
pos	.5	label's position

33.1.1. Labels above and below a segment



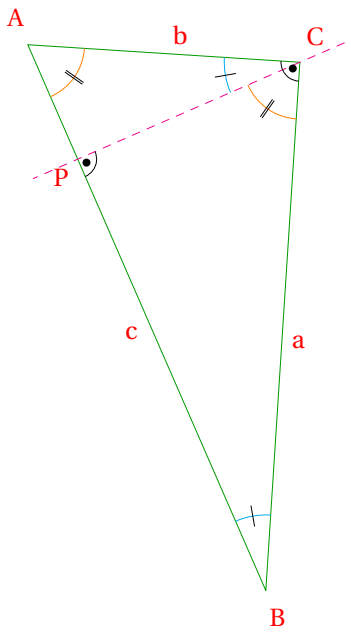

```
\begin{tikzpicture}
\tkzDefPoint(0,0){A}
\tkzDefPoint(6,0){B}
\tkzDrawSegment(A,B)
\tkzLabelSegment[above,pos=.8](A,B){$a$}
\tkzLabelSegment[below,pos=.2](A,B){$4$}
\end{tikzpicture}
```

33.1.2. Example: blackboard




```
\tikzstyle{background rectangle}=[fill=black]
\begin{tikzpicture}[show background rectangle,scale=.4]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(1,0){I}
  \tkzDefPoint(10,0){A}
  \tkzDefPointWith[orthogonal normed,K=4](I,A)
   \tkzGetPoint{H}
  \tkzDefMidPoint(O,A) \tkzGetPoint{M}
  \tkzInterLC(I,H)(M,A)\tkzGetPoints{B}{C}
  \tkzDrawSegments[color=white,line width=1pt](I,H O,A)
  \tkzDrawPoints[color=white](O,I,A,B,M)
  \tkzMarkRightAngle[color=white,line width=1pt](A,I,B)
  \tkzDrawArc[color=white,line width=1pt,
              style=dashed](M,A)(O)
  \tkzLabelSegment[white,right=1ex,pos=.5](I,B){$\sqrt{a}$}
  \tkzLabelSegment[white,below=1ex,pos=.5](O,I){$1$}
  \tkzLabelSegment[pos=.6,white,below=1ex](I,A){$a$}
\end{tikzpicture}
```

33.1.3. Labels and option swap

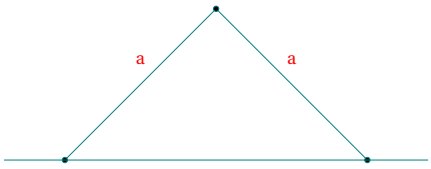



```
\begin{tikzpicture}[rotate=-60]
\tkzSetUpStyle[red,auto]{label style}
\tkzDefPoint(0,1){A}
\tkzDefPoint(2,4){C}
\tkzDefPointWith[orthogonal normed,K=7](C,A)
\tkzGetPoint{B}
\tkzDefSpcTriangle[orthic](A,B,C){N,O,P}
\tkzDefTriangleCenter[circum](A,B,C)
\tkzGetPoint{O}
\tkzDrawPolygon[green!60!black](A,B,C)
\tkzDrawLine[dashed,color=magenta](C,P)
\tkzLabelSegment(B,A){$c$}
\tkzLabelSegment[swap](B,C){$a$}
\tkzLabelSegment[swap](C,A){$b$}
\tkzMarkAngles[size=1,
     color=cyan,mark=|](C,B,A A,C,P)
\tkzMarkAngle[size=0.75,
     color=orange,mark=||](P,C,B)
\tkzMarkAngle[size=0.75,
      color=orange,mark=||](B,A,C)
\tkzMarkRightAngles[german](A,C,B B,P,C)
\tkzAutoLabelPoints[center = O,dist= .1](A,B,C)
 \tkzLabelPoint[below left](P){$P$}
 \end{tikzpicture}
```

\tkzLabelSegments[⟨local options⟩](⟨pt1,pt2 pt3,pt4 ...⟩)

The arguments are a list of pairs of points. The Ti*k*Z styles are available.

33.1.4. Labels for an isosceles triangle




```
\begin{tikzpicture}[scale=1]
 \tkzDefPoints{0/0/O,2/2/A,4/0/B,6/2/C}
 \tkzDrawSegments(O,A A,B)
 \tkzDrawPoints(O,A,B)
 \tkzDrawLine(O,B)
 \tkzLabelSegments[color=red,above=4pt](O,A A,B){$a$}
\end{tikzpicture}
```

34. Labelling lines

34.1. The macro \tkzLabelLine

\tkzLabelLine[⟨local options⟩](⟨pt1,pt2⟩){⟨label⟩}

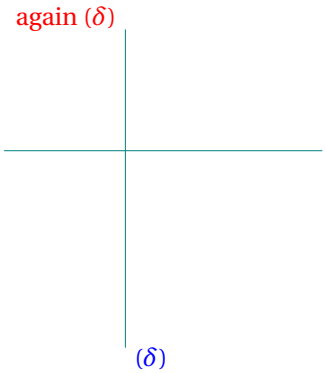
arguments	default	definition
label		\tkzLabelLine(A,B){Δ}

options	default	definition
pos	.5	**pos** is an option for TikZ, but essential in this case…

As options, in addition to **pos**, you can use all the TikZ styles, in particular placements such as **above**, **right**, …

34.1.1. Example with \tkzLabelLine

An important option is **pos**: it places the label along the line. The value of **pos** can be greater than 1 or negative.




```
\begin{tikzpicture}
   \tkzDefPoints{0/0/A,3/0/B,1/1/C}
   \tkzDefLine[perpendicular=through C,K=-1](A,B)
   \tkzGetPoint{c}
   \tkzDrawLines(A,B C,c)
   \tkzLabelLine[pos=1.25,blue,right](C,c){$(\delta)$}
   \tkzLabelLine[pos=-0.25,red,left](C,c){again $(\delta)$}
\end{tikzpicture}
```

35. Labelling angles

35.1. The macros \tkzLabelAngle and \tkzLabelAngles

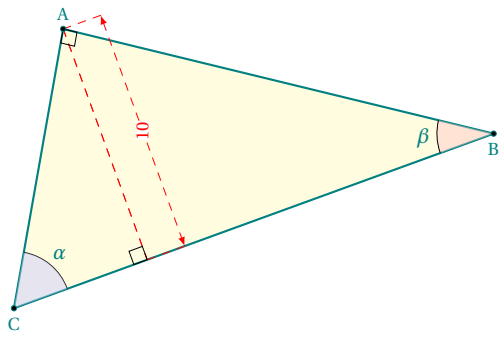
\tkzLabelAngle[⟨local options⟩](⟨A,O,B⟩)

There is only one option, **dist** (with or without unit), which can be replaced by the TikZ option **pos** (without unit). By default, the value is in centimeters.

options	default	definition
pos	1	or dist, controls the distance from the top to the label.

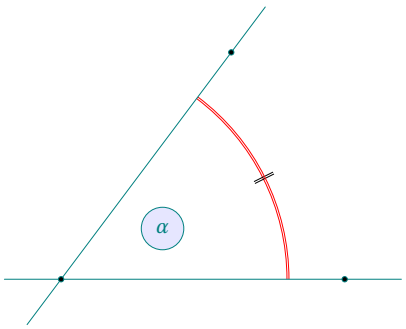
You can move the label with all the TikZ options: rotate, shift, below, etc.

35.1.1. Example by J. S. Bibra (TEX.SE)



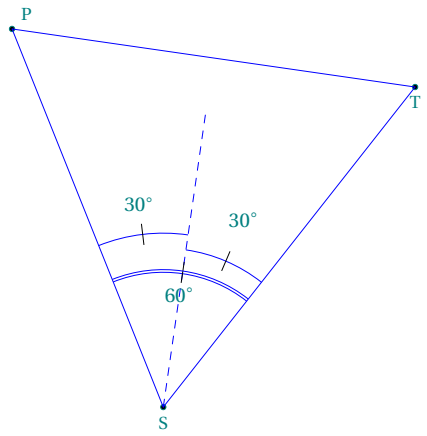

```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoint(0,0){C}
  \tkzDefPoint(20:9){B}
  \tkzDefPoint(80:5){A}
  \tkzDefPointsBy[projection=onto B--C](A){a}
  \tkzDrawPolygon[thick,fill=yellow!15](A,B,C)
  \tkzDrawSegment[dashed, red](A,a)
  \tkzDrawSegment[style=red, dashed,
  dim={$10$,15pt,midway,font=\scriptsize,
   rotate=90}](A,a)
  \tkzMarkAngle(B,C,A)
  \tkzMarkRightAngle(A,a,C)
  \tkzMarkRightAngle(C,A,B)
  \tkzFillAngle[fill=blue!20, opacity=0.5](B,C,A)
  \tkzFillAngle[fill=red!20, opacity=0.5](A,B,C)
  \tkzLabelAngle[pos=1.25](A,B,C){$\beta$}
  \tkzLabelAngle[pos=1.25](B,C,A){$\alpha$}
  \tkzMarkAngle(A,B,C)
  \tkzDrawPoints(A,B,C)
  \tkzLabelPoints(B,C)
  \tkzLabelPoints[above](A)
\end{tikzpicture}
```

35.1.2. With pos




```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoints{0/0/O,5/0/A,3/4/B}
  \tkzMarkAngle[size = 4,mark = ||,
      arc=ll,color = red](A,O,B)%
  \tkzDrawLines(O,A O,B)
  \tkzDrawPoints(O,A,B)
  \tkzLabelAngle[pos=2,draw,circle,
      fill=blue!10](A,O,B){$\alpha$}
\end{tikzpicture}
```

35.1.3. pos and \tkzLabelAngles




```
\begin{tikzpicture}[rotate=30]
  \tkzDefPoint(2,1){S}
  \tkzDefPoint(7,3){T}
  \tkzDefPointBy[rotation=center S angle 60](T)
  \tkzGetPoint{P}
  \tkzDefLine[bisector,normed](T,S,P)
  \tkzGetPoint{s}
  \tkzDrawPoints(S,T,P)
  \tkzDrawPolygon[color=blue](S,T,P)
  \tkzDrawLine[dashed,color=blue,add=0 and 3](S,s)
  \tkzLabelPoint[above right](P){$P$}
  \tkzLabelPoints(S,T)
  \tkzMarkAngle[size = 1.8,mark = |,arc=ll,
                    color = blue](T,S,P)
  \tkzMarkAngle[size = 2.1,mark = |,arc=l,
                    color = blue](T,S,s)
  \tkzMarkAngle[size = 2.3,mark = |,arc=l,
                    color = blue](s,S,P)
 \tkzLabelAngle[pos = 1.5](T,S,P){$60^{\circ}$}%
 \tkzLabelAngles[pos = 2.7](T,S,s s,S,P){%
                          $30^{\circ}$}%
\end{tikzpicture}
```

\tkzLabelAngles[⟨local options⟩](⟨A,O,B⟩)(⟨A',O',B'⟩)etc.

With common options, there is a macro for several angles.

Finally, we need to give a label to a circle. Several methods are possible; here we look at \tkzLabelCircle.

36. Labelling circles

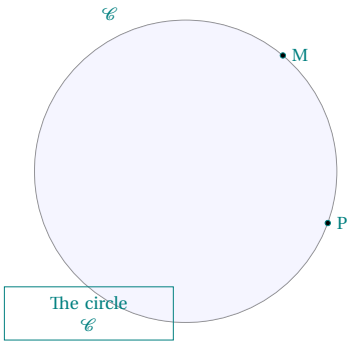
36.1. The macro \tkzLabelCircle

\tkzLabelCircle[⟨tikz options⟩](⟨O,A⟩)(⟨angle⟩){⟨label⟩}

options	default	definition
tikz options		circle O center through A

The TikZ styles can be used. The label is created, so it is passed between braces.

36.1.1. Labels around a circle




```
\begin{tikzpicture}
 \tkzDefPoint(0,0){O} \tkzDefPoint(2,0){N}
 \tkzDefPointBy[rotation=center O angle 50](N)
     \tkzGetPoint{M}
 \tkzDefPointBy[rotation=center O angle -20](N)
      \tkzGetPoint{P}
 \tkzDefPointBy[rotation=center O angle 125](N)
      \tkzGetPoint{P'}
 \tkzLabelCircle[above=4pt](O,N)(120){$\mathcal{C}$}
 \tkzDrawCircle(O,M)
 \tkzFillCircle[color=blue!10,opacity=.4](O,M)
 \tkzLabelCircle[draw,
       text width=2cm,text centered,left=24pt](O,M)(-120)%
          {The circle\\ $\mathcal{C}$}
 \tkzDrawPoints(M,P)\tkzLabelPoints[right](M,P)
\end{tikzpicture}
```

37. Labelling arcs

37.1. The macro \tkzLabelArc

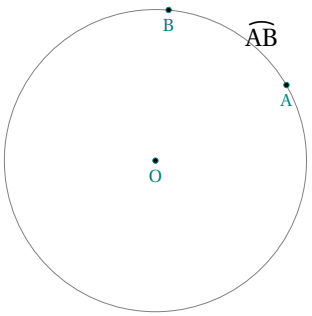
\tkzLabelArc[⟨local options⟩](⟨pt1,pt2,pt3⟩){⟨label⟩}

This macro places a label along an arc. The options are those of Ti*k*Z, for example **pos**.

argument	example	definition
label	\tkzLabelArc(A,B){5}	label text
(pt1,pt2,pt3)	(O,A,B)	label along the arc $\overset{\frown}{AB}$

options	default	definition
pos	.5	label's position

37.1.1. Label on arc




```
\begin{tikzpicture}
\tkzDefPoint(0,0){O}
\pgfmathsetmacro\r{2}
\tkzDefPoint(30:\r){A}
\tkzDefPoint(85:\r){B}
\tkzDrawCircle(O,A)
\tkzDrawPoints(B,A,O)
\tkzLabelArc[right=2pt](O,A,B){$\widearc{AB}$}
\tkzLabelPoints(A,B,O)
\end{tikzpicture}
```

Part VIII.

Complements

38. Using the compass

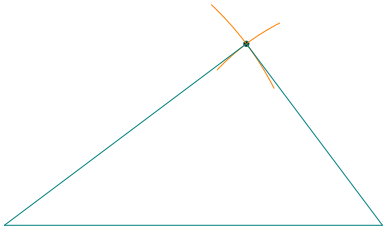
38.1. Main macro \tkzCompass

\tkzCompass[⟨local options⟩](⟨A,B⟩)

This macro leaves a compass mark, i.e. an arc through a given point; the center must be given. Several specific options change the appearance of the arc, as well as Ti*k*Z options such as style, color, line width, etc.
You can set the length of the arc with the option `length` or the option `delta`.

options	default	definition
delta	0 (deg)	Increases the angle of the arc symmetrically
length	1 (cm)	Changes the length (in cm)

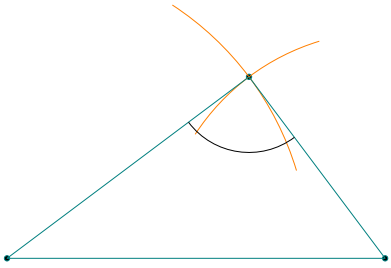
38.1.1. Option length



```
\begin{tikzpicture}
  \tkzDefPoint(1,1){A}
  \tkzDefPoint(6,1){B}
  \tkzInterCC[R](A,4)(B,3)
  \tkzGetPoints{C}{D}
  \tkzDrawPoint(C)
  \tkzCompass[length=1.5](A,C)
  \tkzCompass(B,C)
  \tkzDrawSegments(A,B A,C B,C)
\end{tikzpicture}
```

38.1.2. Option delta

delta is an angle in degrees: the mark goes from $-\delta$ to $+\delta$ around the point. Decimal values are accepted. With the default value 0, the length of the mark is given by the option **length**.



```
\begin{tikzpicture}
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(5,0){B}
  \tkzInterCC[R](A,4)(B,3)
  \tkzGetPoints{C}{D}
  \tkzDrawPoints(A,B,C)
  \tkzCompass[delta=20](A,C)
  \tkzCompass[delta=20](B,C)
  \tkzDrawPolygon(A,B,C)
  \tkzMarkAngle(A,C,B)
\end{tikzpicture}
```

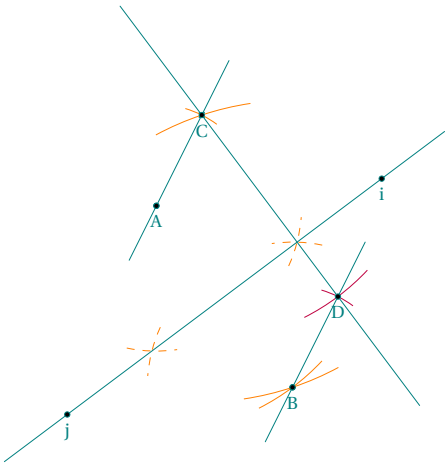
38.2. Multiple constructions \tkzCompasss

\tkzCompasss[⟨local options⟩](⟨pt1,pt2 pt3,pt4,…⟩)

Note that the arguments are lists of pairs of points. This saves a few lines of code.

options	default	definition
delta	0	Modifies the angle of the arc by increasing it symmetrically
length	1	Changes the length

38.2.1. Use of \tkzCompasss




```
\begin{tikzpicture}[scale=.6]
 \tkzDefPoint(2,2){A}  \tkzDefPoint(5,-2){B}
 \tkzDefPoint(3,4){C}  \tkzDrawPoints(A,B)
 \tkzDrawPoint[shape=cross out](C)
 \tkzCompasss[new](A,B A,C B,C C,B)
 \tkzShowLine[mediator,new,dashed,length = 2](A,B)
 \tkzShowLine[parallel = through C,
                    color=purple,length=2](A,B)
 \tkzDefLine[mediator](A,B)
  \tkzGetPoints{i}{j}
 \tkzDefLine[parallel=through C](A,B)
   \tkzGetPoint{D}
 \tkzDrawLines[add=.6 and .6](C,D A,C B,D)
 \tkzDrawLines(i,j) \tkzDrawPoints(A,B,C,i,j,D)
 \tkzLabelPoints(A,B,C,i,j,D)
\end{tikzpicture}
```

39. The Show

39.1. Show the constructions of some lines \tkzShowLine

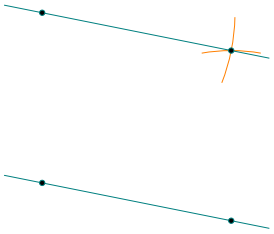
\tkzShowLine[⟨local options⟩](⟨pt1,pt2⟩) or (⟨pt1,pt2,pt3⟩)

These constructions concern mediators (perpendicular bisectors), perpendicular or parallel lines through a given point, and bisectors. The arguments are therefore lists of two or three points. Several options adjust the constructions. The idea of this macro comes from Yves Combe.

options	default	definition
mediator	mediator	displays the constructions of a mediator
perpendicular	mediator	constructions for a perpendicular
orthogonal	mediator	idem
bisector	mediator	constructions for a bisector
K	1	circle within a triangle
length	1	in cm, length of a arc
ratio	.5	arc length ratio
gap	2	placing the point of construction
size	1	radius of an arc (see bisector)

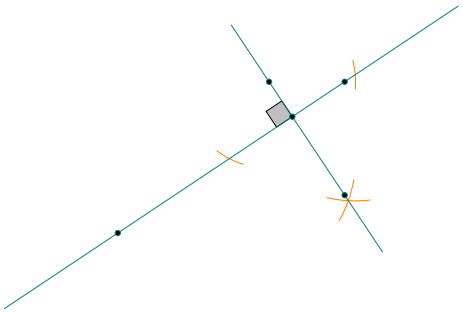
Of course, you can add all the Ti*k*Z styles.

39.1.1. Example of \tkzShowLine and parallel



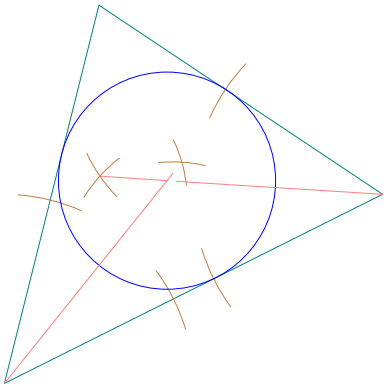
```
\begin{tikzpicture}
 \tkzDefPoints{-1.5/-0.25/A,1/-0.75/B,-1.5/2/C}
 \tkzDrawLine(A,B)
 \tkzDefLine[parallel=through C](A,B) \tkzGetPoint{c}
 \tkzShowLine[parallel=through C](A,B)
 \tkzDrawLine(C,c) \tkzDrawPoints(A,B,C,c)
\end{tikzpicture}
```

39.1.2. Example of \tkzShowLine and perpendicular



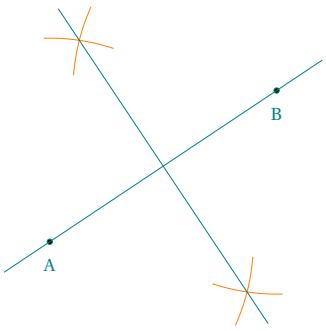
```
\begin{tikzpicture}
\tkzDefPoints{0/0/A, 3/2/B, 2/2/C}
\tkzDefLine[perpendicular=through C,K=-.5](A,B)
\tkzGetPoint{c}
\tkzShowLine[perpendicular=through C,K=-.5,gap=3](A,B)
\tkzDefPointBy[projection=onto A--B](c)
\tkzGetPoint{h}
\tkzMarkRightAngle[fill=lightgray](A,h,C)
\tkzDrawLines[add=.5 and .5](A,B C,c)
\tkzDrawPoints(A,B,C,h,c)
\end{tikzpicture}
```

39.1.3. Example of \tkzShowLine and bisector



```
\begin{tikzpicture}[scale=1.25]
 \tkzDefPoints{0/0/A, 4/2/B, 1/4/C}
 \tkzDrawPolygon(A,B,C)
 \tkzSetUpCompass[color=brown,line width=.1 pt]
 \tkzDefLine[bisector](B,A,C)  \tkzGetPoint{a}
 \tkzDefLine[bisector](C,B,A)  \tkzGetPoint{b}
 \tkzInterLL(A,a)(B,b) \tkzGetPoint{I}
 \tkzDefPointBy[projection = onto A--B](I)
   \tkzGetPoint{H}
 \tkzShowLine[bisector,size=2,gap=3,blue](B,A,C)
 \tkzShowLine[bisector,size=2,gap=3,blue](C,B,A)
 \tkzDrawCircle[color=blue,%
 line width=.2pt](I,H)
 \tkzDrawSegments[color=red!50](I,tkzPointResult)
 \tkzDrawLines[add=0 and -0.3,color=red!50](A,a B,b)
\end{tikzpicture}
```

39.1.4. Example of \tkzShowLine and mediator




```
\begin{tikzpicture}
\tkzDefPoint(2,2){A}
\tkzDefPoint(5,4){B}
\tkzDrawPoints(A,B)
\tkzShowLine[mediator,color=orange,length=1](A,B)
\tkzGetPoints{i}{j}
\tkzDrawLines[add=-0.1 and -0.1](i,j)
\tkzDrawLines(A,B)
\tkzLabelPoints[below =3pt](A,B)
\end{tikzpicture}
```

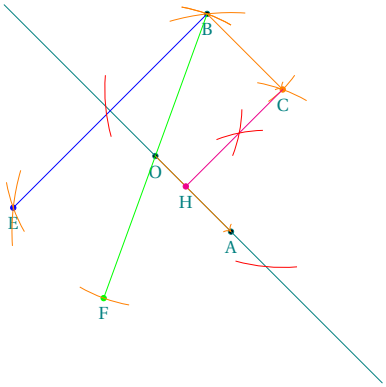
39.2. Constructions of certain transformations \tkzShowTransformation

\tkzShowTransformation[⟨local options⟩](⟨pt1,pt2⟩) or (⟨pt1,pt2,pt3⟩)

These constructions concern orthogonal symmetries, central symmetries, orthogonal projections and translations. Several options adjust the constructions. The idea of this macro comes from Yves Combe.

options	default	definition
reflection= over pt1--pt2	reflection	constructions of orthogonal symmetry
symmetry=center pt	reflection	constructions of central symmetry
projection=onto pt1--pt2	reflection	constructions of a projection
translation=from pt1 to pt2	reflection	constructions of a translation
K	1	circle within a triangle
length	1	arc length
ratio	.5	arc length ratio
gap	2	placing the point of construction
size	1	radius of an arc (see bisector)

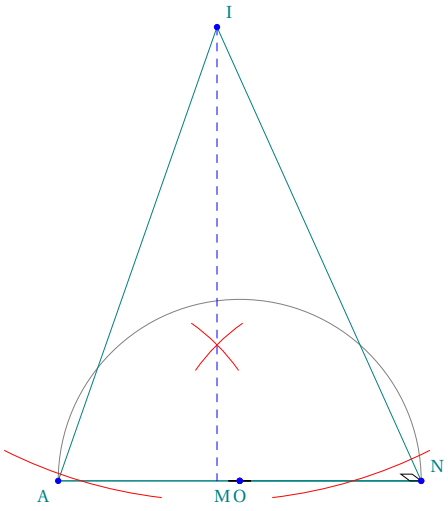
39.2.1. Example of the use of \tkzShowTransformation




```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoint(0,0){O} \tkzDefPoint(2,-2){A}
  \tkzDefPoint(70:4){B} \tkzDrawPoints(A,O,B)
  \tkzLabelPoints(A,O,B)
  \tkzDrawLine[add= 2 and 2](O,A)
  \tkzDefPointBy[translation=from O to A](B)
  \tkzGetPoint{C}
  \tkzDrawPoint[color=orange](C)  \tkzLabelPoints(C)
  \tkzShowTransformation[translation=from O to A,%
            length=2](B)
  \tkzDrawSegments[->,color=orange](O,A B,C)
  \tkzDefPointBy[reflection=over O--A](B) \tkzGetPoint{E}
  \tkzDrawSegment[blue](B,E)
  \tkzDrawPoint[color=blue](E)\tkzLabelPoints(E)
  \tkzShowTransformation[reflection=over O--A,size=2](B)
  \tkzDefPointBy[symmetry=center O](B) \tkzGetPoint{F}
  \tkzDrawSegment[color=green](B,F)
  \tkzDrawPoint[color=green](F)\tkzLabelPoints(F)
  \tkzShowTransformation[symmetry=center O,%
                      length=2](B)
  \tkzDefPointBy[projection=onto O--A](C)
  \tkzGetPoint{H}
  \tkzDrawSegments[color=magenta](C,H)
  \tkzDrawPoint[color=magenta](H)\tkzLabelPoints(H)
  \tkzShowTransformation[projection=onto O--A,%
                          color=red,size=3,gap=-2](C)
\end{tikzpicture}
```

39.2.2. Another example of the use of \tkzShowTransformation

You will find this figure again later, without the construction lines.




```
\begin{tikzpicture}[scale=.6]
  \tkzDefPoints{0/0/A,8/0/B,3.5/10/I}
  \tkzDefMidPoint(A,B) \tkzGetPoint{O}
  \tkzDefPointBy[projection=onto A--B](I)
     \tkzGetPoint{J}
  \tkzInterLC(I,A)(O,A)  \tkzGetPoints{M}{M'}
  \tkzInterLC(I,B)(O,A)  \tkzGetPoints{N}{N'}
  \tkzDefMidPoint(A,B) \tkzGetPoint{M}
  \tkzDrawSemiCircle(M,B)
  \tkzDrawSegments(I,A I,B A,B B,M A,N)
  \tkzMarkRightAngles(A,M,B A,N,B)
  \tkzDrawSegment[style=dashed,color=blue](I,J)
  \tkzShowTransformation[projection=onto A--B,
                color=red,size=3,gap=-3](I)
  \tkzDrawPoints[color=red](M,N)
  \tkzDrawPoints[color=blue](O,A,B,I,M)
  \tkzLabelPoints(O)
  \tkzLabelPoints[above right](N,I)
  \tkzLabelPoints[below left](M,A)
\end{tikzpicture}
```

40. Protractor

Based on an idea of Yves Combe, the following macro draws a protractor. The principle is simple: just name a ray. The protractor is placed at its origin O, and the direction of the ray is given by A. The angle is measured counterclockwise; with the option **return**, the protractor is reversed and the angle is measured clockwise.

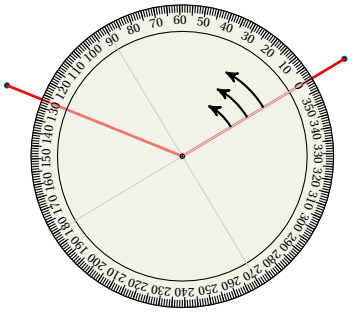
40.1. The macro \tkzProtractor

\tkzProtractor[⟨local options⟩](⟨O,A⟩)

options	default	definition
lw	0.4 pt	line thickness
scale	1	ratio: adjusts the size of the protractor
return	false	trigonometric circle indirect

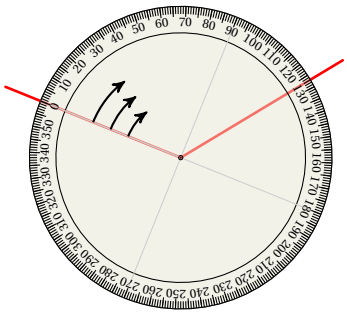
40.1.1. The circular protractor

Measuring counterclockwise:



```
\begin{tikzpicture}[scale=.5]
\tkzDefPoint(2,0){A}\tkzDefPoint(0,0){O}
\tkzDefShiftPoint[A](31:5){B}
\tkzDefShiftPoint[A](158:5){C}
\tkzDrawPoints(A,B,C)
\tkzDrawSegments[color = red,
    line width = 1pt](A,B A,C)
  \tkzProtractor[scale = 1](A,B)
\end{tikzpicture}
```

40.1.2. The circular protractor, transparent and returned



```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoint(2,3){A}
  \tkzDefShiftPoint[A](31:5){B}
   \tkzDefShiftPoint[A](158:5){C}
  \tkzDrawSegments[color=red,line width=1pt](A,B A,C)
  \tkzProtractor[return](A,C)
\end{tikzpicture}
```

41. Miscellaneous and mathematical tools

41.1. Duplicate a segment

The aim is to construct, on a given ray, a segment of the same length as a given segment.

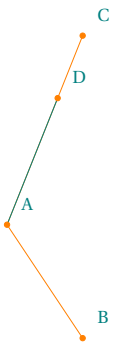
\tkzDuplicateSegment(⟨pt1,pt2⟩)(⟨pt3,pt4⟩){⟨pt5⟩}

This macro constructs, on a given ray, a segment of the same length as a given segment. In fact it defines a point. **\tkzDuplicateSegment** is the new name of **\tkzDuplicateLen**.

arguments	example	explanation
(pt1,pt2)(pt3,pt4){pt5}	**\tkzDuplicateSegment**(A,B)(E,F){C}	AC=EF and C ∈ [AB)

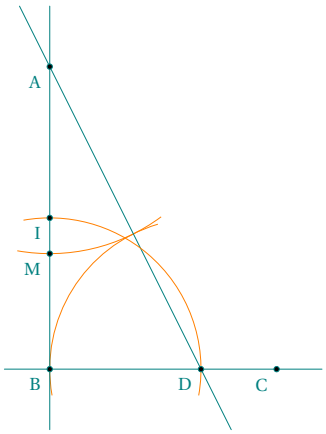
The macro ***\tkzDuplicateLength*** *is identical to this one.*

41.1.1. Use of \tkzDuplicateSegment



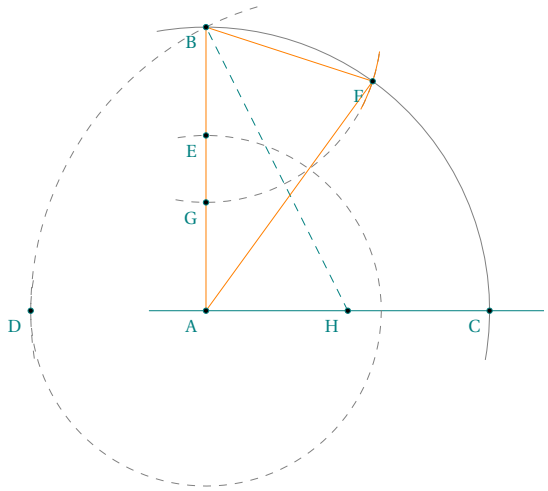

```
\begin{tikzpicture}[scale=.5]
 \tkzDefPoints{0/0/A,2/-3/B,2/5/C}
 \tkzDuplicateSegment(A,B)(A,C)
 \tkzGetPoint{D}
 \tkzDrawSegments[new](A,B A,C)
 \tkzDrawSegment[teal](A,D)
 \tkzDrawPoints[new](A,B,C,D)
 \tkzLabelPoints[above right=3pt](A,B,C,D)
\end{tikzpicture}
```

41.1.2. Golden ratio with \tkzDuplicateSegment




```
\begin{tikzpicture}[rotate=-90,scale=.4]
 \tkzDefPoints{0/0/A,10/0/B}
 \tkzDefMidPoint(A,B)
 \tkzGetPoint{I}
 \tkzDefPointWith[orthogonal,K=-.75](B,A)
 \tkzGetPoint{C}
 \tkzInterLC(B,C)(B,I)  \tkzGetSecondPoint{D}
 \tkzDuplicateSegment(B,D)(D,A) \tkzGetPoint{E}
 \tkzInterLC(A,B)(A,E)   \tkzGetPoints{N}{M}
 \tkzDrawArc[orange,delta=10](D,E)(B)
 \tkzDrawArc[orange,delta=10](A,M)(E)
 \tkzDrawLines(A,B B,C A,D)
 \tkzDrawArc[orange,delta=10](B,D)(I)
 \tkzDrawPoints(A,B,D,C,M,I)
 \tkzLabelPoints[below left](A,B,D,C,M,I)
\end{tikzpicture}
```

41.1.3. Golden triangle or sublime triangle




```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoints{0/0/A,5/0/C,0/5/B}
  \tkzDefMidPoint(A,C)\tkzGetPoint{H}
  \tkzDuplicateSegment(H,B)(H,A)\tkzGetPoint{D}
  \tkzDuplicateSegment(A,D)(A,B)\tkzGetPoint{E}
  \tkzDuplicateSegment(A,D)(B,A)\tkzGetPoint{G}
  \tkzInterCC(A,C)(B,G)\tkzGetSecondPoint{F}
  \tkzDrawLine(A,C)
  \tkzDrawArc(A,C)(B)
  \begin{scope}[arc style/.style={color=gray,%
                                  style=dashed}]
    \tkzDrawArc(H,B)(D)
    \tkzDrawArc(A,D)(B)
    \tkzDrawArc(B,G)(F)
  \end{scope}
  \tkzDrawSegment[dashed](H,B)
  \tkzCompass(B,F)
  \tkzDrawPolygon[new](A,B,F)
  \tkzDrawPoints(A,...,H)
  \tkzLabelPoints[below left](A,...,H)
\end{tikzpicture}
```

41.2. Segment length \tkzCalcLength

Ti*k*Z has a function named **veclen**. It calculates AB if A and B are two points.
The only problem is that the Ti*k*Z version is not accurate enough in some cases. My version uses the package `xfp`: it is slower, but more accurate.

\tkzCalcLength[⟨local options⟩](⟨pt1,pt2⟩)

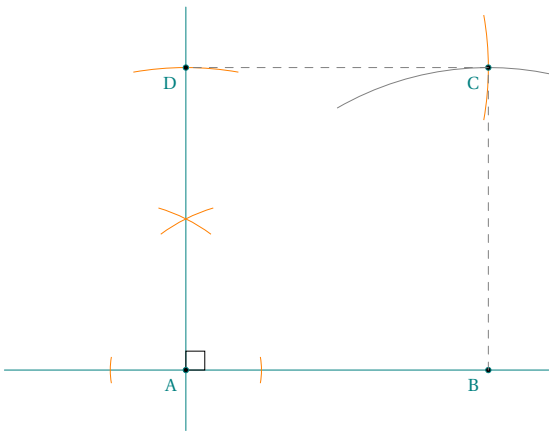
You can store the result with the macro **\tkzGetLength**; for example **\tkzGetLength{dAB}** defines the macro **\dAB**.

arguments	example	explanation
(pt1,pt2){name of macro}	**\tkzCalcLength**(A,B)	**\dAB** gives AB in cm

There is only one option.

options	default	example
cm	true	\tkzCalcLength(A,B) After \tkzGetLength{dAB} \dAB gives AB in cm

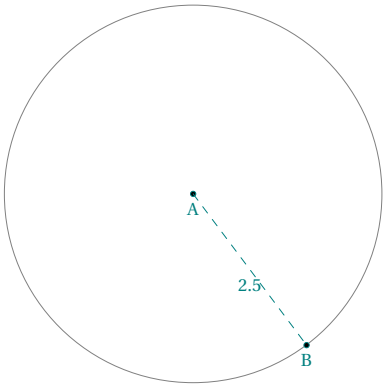
41.2.1. Compass square construction




```
\begin{tikzpicture}[scale=1]
  \tkzDefPoint(0,0){A} \tkzDefPoint(4,0){B}
  \tkzCalcLength(A,B)\tkzGetLength{dAB}
  \tkzDefLine[perpendicular=through A](A,B)
  \tkzGetPoint{D}
  \tkzDefPointWith[orthogonal,K=-1](B,A)
    \tkzGetPoint{F}
  \tkzGetPoint{C}
  \tkzDrawLine[add= .6 and .2](A,B)
  \tkzDrawLine(A,D)
  \tkzShowLine[orthogonal=through A,gap=2](A,B)
  \tkzMarkRightAngle(B,A,D)
  \tkzCompasss(A,D D,C)
  \tkzDrawArc[R](B,\dAB)(80,110)
  \tkzDrawPoints(A,B,C,D)
  \tkzDrawSegments[color=gray,style=dashed](B,C C,D)
  \tkzLabelPoints[below left](A,B,C,D)
\end{tikzpicture}
```

41.2.2. Radius of a circle defined by a diameter

The macro \tkzDefCircle[radius](A,B) defines a radius, which we retrieve with \tkzGetLength; the result is in cm.




```
\begin{tikzpicture}[scale=.5]
 \tkzDefPoint(0,0){A}
 \tkzDefPoint(3,-4){B}
 \tkzDefMidPoint(A,B) \tkzGetPoint{M}
 \tkzCalcLength(M,B)\tkzGetLength{rAB}
 \tkzDrawCircle(A,B)
 \tkzDrawPoints(A,B)
 \tkzLabelPoints(A,B)
 \tkzDrawSegment[dashed](A,B)
 \tkzLabelSegment(A,B){$\pgfmathprintnumber{\rAB}$}
\end{tikzpicture}
```

41.3. Transformation from pt to cm or cm to pt

These macros are simple: a division or a multiplication by 28.45274. They are:

\tkzpttocm(⟨number⟩){⟨name of macro⟩}

The result is stored in a macro.

arguments	example	explanation
(number){name of macro}	\tkzpttocm(120){len}	\len gives a number of tkznamecm

You have to use \len with the unit cm.

41.4. Change of unit

\tkzcmtopt(⟨number⟩){⟨name of macro⟩}

The result is stored in a macro.

arguments	example	explanation
(number){name of macro}	\tkzcmtopt(5){len}	\len length in pts

The result can be used with \len pt

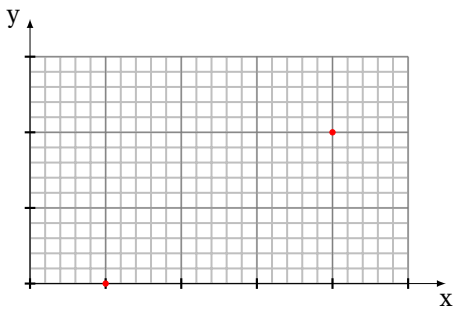
41.5. Get point coordinates

\tkzGetPointCoord(⟨A⟩){⟨name of macro⟩}

arguments	example	explanation
(point){name of macro}	\tkzGetPointCoord(A){A}	\Ax and \Ay give coordinates for A

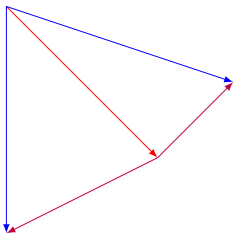
Stores the coordinates of a point in two macros. If the name given is p*, then* \px *and* \py *give the coordinates of the point, in cm.*

41.5.1. Coordinate transfer with \tkzGetPointCoord




```
\begin{tikzpicture}
 \tkzInit[xmax=5,ymax=3]
 \tkzGrid[sub,orange]
 \tkzDrawX \tkzDrawY
 \tkzDefPoint(1,0){A}
 \tkzDefPoint(4,2){B}
 \tkzGetPointCoord(A){a}
 \tkzGetPointCoord(B){b}
 \tkzDefPoint(\ax,\ay){C}
 \tkzDefPoint(\bx,\by){D}
 \tkzDrawPoints[color=red](C,D)
\end{tikzpicture}
```

41.5.2. Sum of vectors with \tkzGetPointCoord



```
\begin{tikzpicture}[>=latex]
  \tkzDefPoint(1,4){a}
  \tkzDefPoint(3,2){b}
  \tkzDefPoint(1,1){c}
  \tkzDrawSegment[->,red](a,b)
  \tkzGetPointCoord(c){c}
  \draw[color=blue,->](a) -- ([shift=(b)]\cx,\cy) ;
  \draw[color=purple,->](b) -- ([shift=(b)]\cx,\cy) ;
  \tkzDrawSegment[->,blue](a,c)
  \tkzDrawSegment[->,purple](b,c)
\end{tikzpicture}
```

41.6. Swap labels of points

`\tkzSwapPoints(⟨pt1,pt2⟩)`

arguments	example	explanation
(pt1,pt2)	`\tkzSwapPoints(A,B)`	now A has the coordinates of B

The points have exchanged their coordinates.

41.6.1. Use of \tkzSwapPoints




```
\begin{tikzpicture}
  \tkzDefPoints{0/0/O,5/-1/A,2/2/B}
   \tkzSwapPoints(A,B)
   \tkzDrawPoints(O,A,B)
   \tkzLabelPoints(O,A,B)
\end{tikzpicture}
```

41.7. Dot Product

In Euclidean geometry, the dot product of two vectors, computed from their Cartesian coordinates, is widely used.

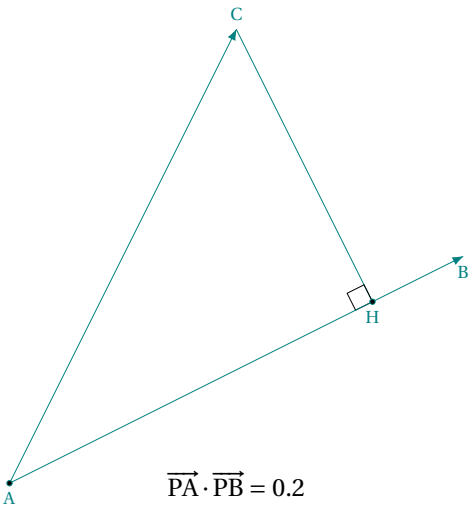
`\tkzDotProduct(⟨pt1,pt2,pt3⟩)`

The dot product of two vectors $\vec{u} = [a,b]$ and $\vec{v} = [a',b']$ is defined as: $\vec{u}\cdot\vec{v} = aa' + bb'$
$\vec{u} = \overrightarrow{pt1pt2}$ $\vec{v} = \overrightarrow{pt1pt3}$

arguments	example	explanation
(pt1,pt2,pt3)	`\tkzDotProduct(A,B,C)`	the result is $\overrightarrow{AB}\cdot\overrightarrow{AC}$

The result is a number that can be retrieved with ***\tkzGetResult****.*

41.7.1. Simple example

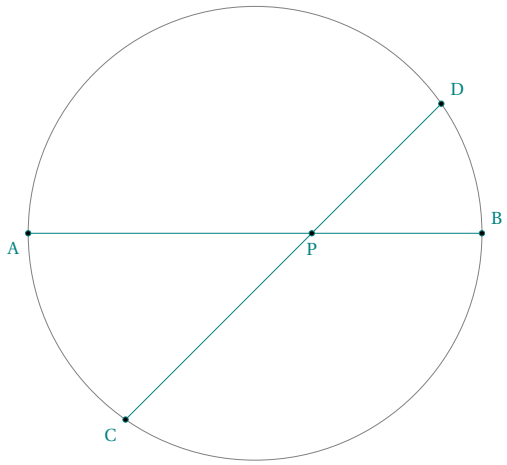


$$\overrightarrow{PA}\cdot\overrightarrow{PB} = 0.2$$

$$PA \times PH = 0.2$$

```
\begin{tikzpicture}
  \tkzDefPoints{-2/-3/A,4/0/B,1/3/C}
  \tkzDefPointBy[projection= onto A--B](C)
  \tkzGetPoint{H}
  \tkzDrawSegment(C,H)
  \tkzMarkRightAngle(C,H,A)
  \tkzDrawSegments[vector style](A,B A,C)
  \tkzDrawPoints(A,H) \tkzLabelPoints(A,B,H)
  \tkzLabelPoints[above](C)
  \tkzDotProduct(A,B,C) \tkzGetResult{pabc}
 % \pgfmathparse{round(10*\pabc)/10}
  \let\pabc\pgfmathresult
  \node at (1,-3) {$\overrightarrow{PA}\cdot \overrightarrow{PB}=\pa
  \tkzDotProduct(A,H,B) \tkzGetResult{phab}
 % \pgfmathparse{round(10*\phab)/10}
  \let\phab\pgfmathresult
  \node at (1,-4) {$PA \times PH = \phab $};
\end{tikzpicture}
```

41.7.2. Cocyclic points



$$\overrightarrow{PA}\cdot\overrightarrow{PB} = \overrightarrow{PC}\cdot\overrightarrow{PD}$$

$$\overrightarrow{PA}\cdot\overrightarrow{PB} = -15.0$$

$$\overrightarrow{PC}\cdot\overrightarrow{PD} = -15.0$$

```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoints{1/2/O,5/2/B,2/2/P,3/3/Q}
  \tkzInterLC[common=B](O,B)(O,B) \tkzGetFirstPoint{A}
  \tkzInterLC[common=B](P,Q)(O,B) \tkzGetPoints{C}{D}
  \tkzDrawCircle(O,B)
  \tkzDrawSegments(A,B C,D)
  \tkzDrawPoints(A,B,C,D,P)
  \tkzLabelPoints(P)
  \tkzLabelPoints[below left](A,C)
  \tkzLabelPoints[above right](B,D)
  \tkzDotProduct(P,A,B) \tkzGetResult{pab}
  \pgfmathparse{round(10*\pab)/10}
  \let\pab\pgfmathresult
  \tkzDotProduct(P,C,D) \tkzGetResult{pcd}
  \pgfmathparse{round(10*\pcd)/10}
  \let\pcd\pgfmathresult
  \node at (1,-3) {%
  $\overrightarrow{PA}\cdot \overrightarrow{PB} =
   \overrightarrow{PC}\cdot \overrightarrow{PD}$};
  \node at (1,-4)%
  {$\overrightarrow{PA}\cdot \overrightarrow{PB}=\pab$};
 \node at (1,-5){%
 $\overrightarrow{PC}\cdot \overrightarrow{PD} =\pcd$};
\end{tikzpicture}
```

41.8. Power of a point with respect to a circle

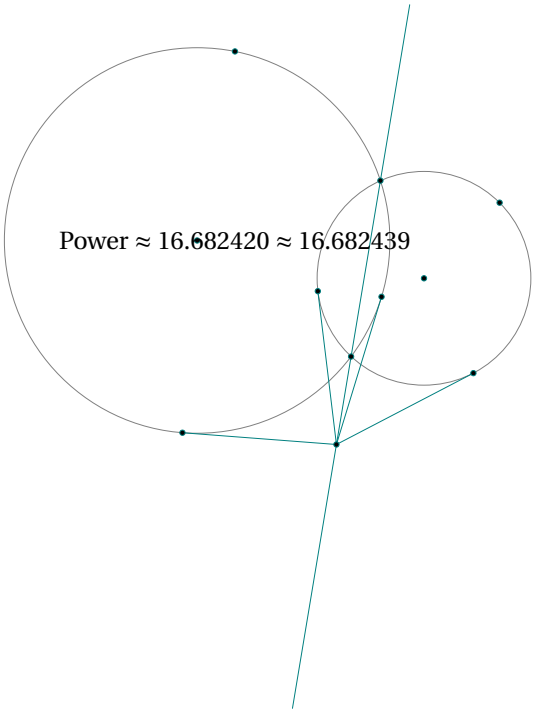
\tkzPowerCircle(⟨pt1⟩)(⟨pt2,pt3⟩)

arguments	example	explanation
(pt1)(pt2,pt3)	\tkzPowerCircle(A)(O,M)	power of A with respect to the circle (O,A)

The result is a number that represents the power of a point with respect to a circle.

41.8.1. Power from the radical axis

In this example, the radical axis (EF) has been drawn. A point H has been chosen on (EF), and the power of H with respect to the circle with center A has been calculated, as well as HS^2. You can check that the power of H with respect to the circle with center C, and HS'^2, HT^2, HT'^2, give the same result.




```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoints{-1/0/A,0/5/B,5/-1/C,7/1/D}
  \tkzDrawCircles(A,B C,D)
  \tkzDefRadicalAxis(A,B)(C,D) \tkzGetPoints{E}{F}
  \tkzDrawLine[add=1 and 2](E,F)
  \tkzDefPointOnLine[pos=1.5](E,F)
  \tkzGetPoint{H}
  \tkzDefLine[tangent from = H](A,B)
  \tkzGetPoints{T}{T'}
  \tkzDefLine[tangent from = H](C,D)
  \tkzGetPoints{S}{S'}
  \tkzDrawSegments(H,T H,T' H,S H,S')
  \tkzDrawPoints(A,B,C,D,E,F,H,T,T',S,S')
  \tkzPowerCircle(H)(A,B) \tkzGetResult{pw}
  \tkzDotProduct(H,S,S) \tkzGetResult{phtt}
  \node {Power $\approx \pw \approx \phtt$};
\end{tikzpicture}
```

41.9. Radical axis

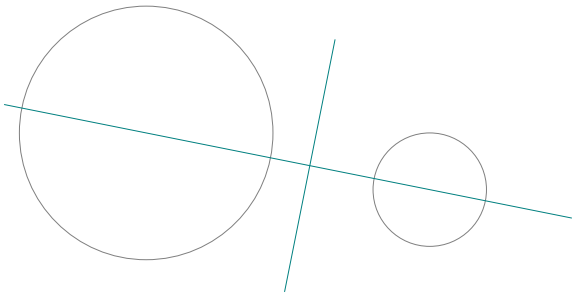
In geometry, the radical axis of two non-concentric circles is the set of points that have the same power with respect to both circles. Here \tkzDefRadicalAxis(A,B)(C,D) gives the radical axis of the two circles $\mathscr{C}$(A,B) and $\mathscr{C}$(C,D).

\tkzDefRadicalAxis(⟨pt1,pt2⟩)(⟨pt3,pt4⟩)

arguments	example	explanation
(pt1,pt2)(pt3,pt4)	\tkzDefRadicalAxis(A,B)(C,D)	Two circles with centers A and C

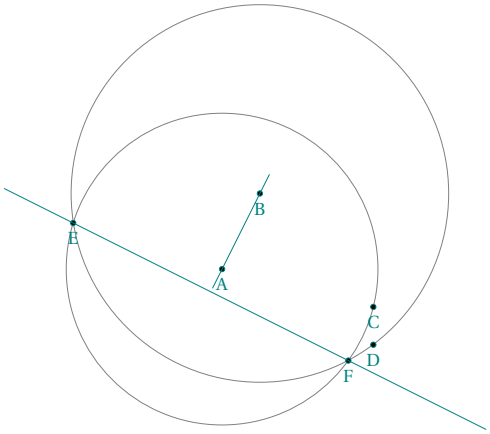
The result is two points of the radical axis.

41.9.1. Two disjoint circles



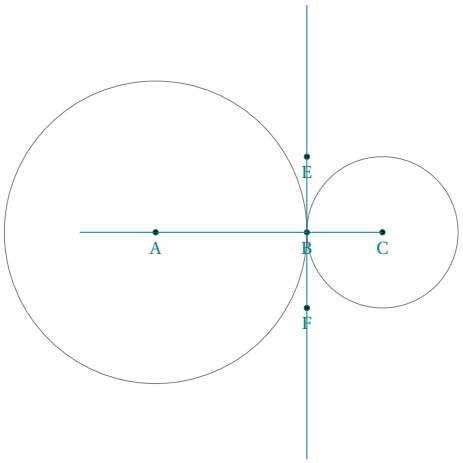
```
\begin{tikzpicture}[scale=.75]
    \tkzDefPoints{-1/0/A,0/2/B,4/-1/C,4/0/D}
    \tkzDrawCircles(A,B C,D)
    \tkzDefRadicalAxis(A,B)(C,D)
     \tkzGetPoints{E}{F}
    \tkzDrawLine[add=1 and 2](E,F)
    \tkzDrawLine[add=.5 and .5](A,C)
\end{tikzpicture}
```

41.9.2. Two intersecting circles



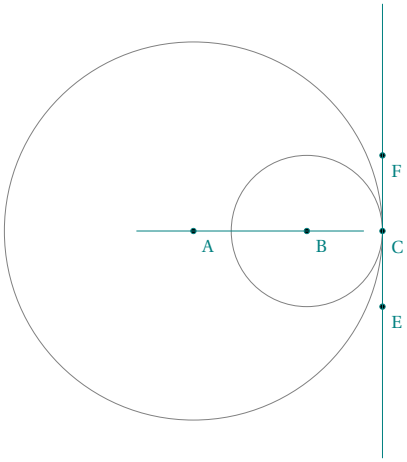

```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoints{-1/0/A,0/2/B,3/-1/C,3/-2/D}
  \tkzDrawCircles(A,C B,D)
  \tkzDefRadicalAxis(A,C)(B,D)
  \tkzGetPoints{E}{F}
  \tkzDrawPoints(A,B,C,D,E,F)
    \tkzLabelPoints(A,B,C,D,E,F)
  \tkzDrawLine[add=.25 and .5](E,F)
  \tkzDrawLine[add=.25 and .25](A,B)
\end{tikzpicture}
```

41.9.3. Two externally tangent circles



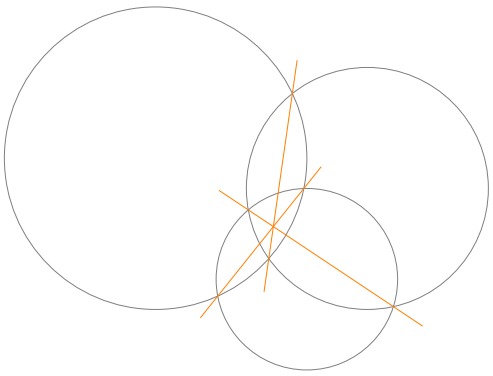

```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoints{0/0/A,4/0/B,6/0/C}
  \tkzDrawCircles(A,B C,B)
  \tkzDefRadicalAxis(A,B)(C,B)
  \tkzGetPoints{E}{F}
  \tkzDrawPoints(A,B,C,E,F)
    \tkzLabelPoints(A,B,C,E,F)
  \tkzDrawLine[add=1 and 1](E,F)
  \tkzDrawLine[add=.5 and .5](A,B)
\end{tikzpicture}
```

41.9.4. Two internally tangent circles




```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoints{0/0/A,3/0/B,5/0/C}
  \tkzDrawCircles(A,C B,C)
  \tkzDefRadicalAxis(A,C)(B,C)
  \tkzGetPoints{E}{F}
  \tkzDrawPoints(A,B,C,E,F)
  \tkzLabelPoints[below right](A,B,C,E,F)
  \tkzDrawLine[add=1 and 1](E,F)
  \tkzDrawLine[add=.5 and .5](A,B)
\end{tikzpicture}
```

41.9.5. Three circles



```
\begin{tikzpicture}[scale=.4]
  \tkzDefPoints{0/0/A,5/0/a,7/-1/B,3/-1/b,5/-4/C,2/-
4/c}
  \tkzDrawCircles(A,a B,b C,c)
  \tkzDefRadicalAxis(A,a)(B,b) \tkzGetPoints{i}{j}
  \tkzDefRadicalAxis(A,a)(C,c) \tkzGetPoints{k}{l}
  \tkzDefRadicalAxis(C,c)(B,b) \tkzGetPoints{m}{n}
  \tkzDrawLines[new](i,j k,l m,n)
\end{tikzpicture}
```

41.10. \tkzIsLinear, \tkzIsOrtho

\tkzIsLinear(⟨pt1,pt2,pt3⟩)

arguments	example	explanation
(pt1,pt2,pt3)	\tkzIsLinear(A,B,C)	A,B,C aligned ?

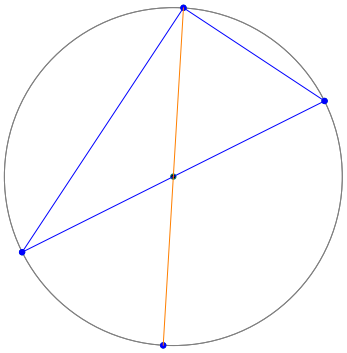
\tkzIsLinear tests whether the three points pt1, pt2, pt3 *are aligned. The test tolerates an angle of about* 0.06°.

\tkzIsOrtho(⟨pt1,pt2,pt3⟩)

arguments	example	explanation
(pt1,pt2,pt3)	\tkzIsOrtho(A,B,C)	(AB) ⊥ (AC) ?

\tkzIsOrtho tests whether the lines (pt1pt2) *and* (pt1pt3) *are orthogonal, with the same tolerance.*

41.10.1. Use of \tkzIsOrtho and \tkzIsLinear



```
\begin{tikzpicture}
\tkzDefPoints{1/-2/A,5/0/B}
\tkzDefCircle[diameter](A,B) \tkzGetPoint{O}
\tkzDrawCircle(O,A)
\tkzDefPointBy[rotation= center O angle 60](B)
\tkzGetPoint{C}
\tkzDefPointBy[rotation= center O angle 60](A)
\tkzGetPoint{D}
\tkzDrawCircle(O,A)
\tkzDrawPoints(A,B,C,D,O)
\tkzIsOrtho(C,A,B)
\iftkzOrtho
  \tkzDrawPolygon[blue](A,B,C)
\tkzDrawPoints[blue](A,B,C,D)
\else
\tkzDrawPoints[red](A,B,C,D)
\fi
 \tkzIsLinear(O,C,D)
  \iftkzLinear
  \tkzDrawSegment[orange](C,D)
  \fi
\end{tikzpicture}
```

Part IX.

Working with styles

42. Predefined styles

How to proceed depends on your use of the package. A good method is to use predefined styles as much as possible, to separate content from form. This is the right method if you plan to create a document with many figures, like this documentation. We will see how to define a global style for a document, and how to use a style locally.
The file **tkz-euclide.cfg** contains the predefined styles of the main objects, the most important being points, lines, segments, circles, arcs and compass marks. If you always use the same styles and create many figures, it is worth creating your own styles. To do this, you need to know which features can be modified, and some notions of Ti*k*Z.
The predefined styles are global: they exist before the figures are created, and it is better not to change them between two figures. On the other hand, they can be modified temporarily inside a figure; they are then defined locally and do not affect the other figures.
Here is how I defined the styles of the document you are reading.

```
\tkzSetUpColors[background=white,text=black]
\tkzSetUpPoint[size=2,color=teal]
\tkzSetUpLine[line width=.4pt,color=teal]
\tkzSetUpCompass[color=orange, line width=.4pt,delta=10]
\tkzSetUpArc[color=gray,line width=.4pt]
\tkzSetUpStyle[orange]{new}
```

The macro **\tkzSetUpColors** sets the background color and the text color. If you don't use it, the colors and fonts of your document are used. Let's see how to define the styles of the main objects.

43. Point style

This is how the points are defined:

```
\tikzset{point style/.style = {%
       draw         = \tkz@euc@pointcolor,
       inner sep    = 0pt,
       shape        = \tkz@euc@pointshape,
       minimum size = \tkz@euc@pointsize,
       fill         = \tkz@euc@pointcolor}}
```

You can of course use **\tikzset**, but the package provides a macro for this: **\tkzSetUpPoint**, which can be used globally or locally.

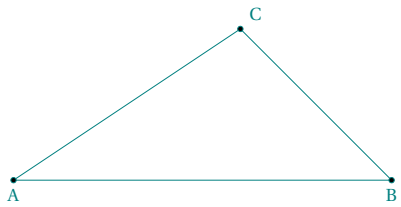
43.1. Use of \tkzSetUpPoint

\tkzSetUpPoint[⟨local options⟩**]**

options	default	definition
color	black	point color
size	3	point size
fill	black!50	inside point color
shape	circle	point shape circle, cross or cross out

43.1.1. Global style or local style

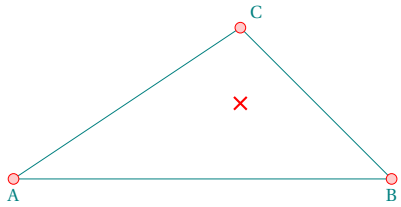
First, here is a figure created with the styles of this documentation; then the style of the points is modified inside the **tikzpicture** environment.
You can use the macro \tkzSetUpPoint globally or locally. If you place it in your preamble or before your first figure, the point style is valid for all the figures of your document. You can use another style locally by placing this command inside a **tikzpicture** environment.




```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,5/0/B,3/2/C,3/1/D}
  \tkzDrawPolygon(A,B,C)
  \tkzDrawPoints(A,B,C)
  \tkzLabelPoints(A,B)
  \tkzLabelPoints[above right](C)
\end{tikzpicture}
```

43.1.2. Local style

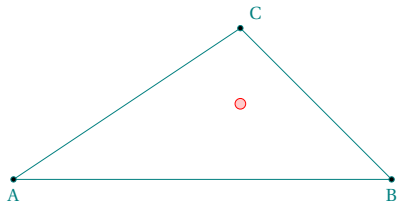
The style of the points is modified locally in the second figure.




```
\begin{tikzpicture}
  \tkzSetUpPoint[size=4,color=red,fill=red!20]
  \tkzDefPoints{0/0/A,5/0/B,3/2/C,3/1/D}
  \tkzDrawPolygon(A,B,C)
  \tkzDrawPoints(A,B,C)
  \tkzDrawPoint[shape=cross out,thick](D)
  \tkzLabelPoints(A,B)
  \tkzLabelPoints[above right](C)
\end{tikzpicture}
```

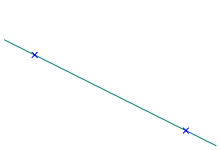
43.1.3. Style and scope

The points get back the initial style. Point D has a new style limited by the environment **scope**. It is also possible to use {...} orThe points get back the initial style. Point D has a new style limited by the environment **scope**. It is also possible to use {...} or \begingoup ... \endgroup.



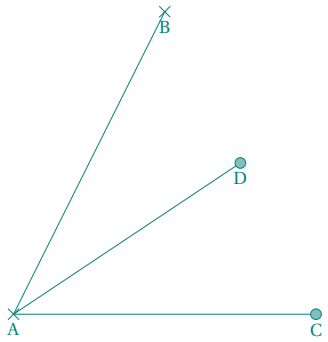

```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,5/0/B,3/2/C,3/1/D}
  \tkzDrawPolygon(A,B,C)
  \tkzDrawPoints(A,B,C)
  \begin{scope}
    \tkzSetUpPoint[size=4,color=red,fill=red!20]
    \tkzDrawPoint(D)
  \end{scope}
  \tkzLabelPoints(A,B)
  \tkzLabelPoints[above right](C)
\end{tikzpicture}
```

43.1.4. Simple example with \tkzSetUpPoint



```
\begin{tikzpicture}
  \tkzSetUpPoint[shape = cross out,color=blue]
  \tkzDefPoint(2,1){A}
  \tkzDefPoint(4,0){B}
  \tkzDrawLine(A,B)
  \tkzDrawPoints(A,B)
\end{tikzpicture}
```

43.1.5. Use of \tkzSetUpPoint inside a group




```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,2/4/B,4/0/C,3/2/D}
  \tkzDrawSegments(A,B A,C A,D)
  {\tkzSetUpPoint[shape=cross out,
            fill= teal!50,
            size=4,color=teal]
  \tkzDrawPoints(A,B)}
  \tkzSetUpPoint[fill= teal!50,size=4,
              color=teal]
   \tkzDrawPoints(C,D)
  \tkzLabelPoints(A,B,C,D)
\end{tikzpicture}
```

44. Line style

There are several ways to change the style of a line: with **\tkzSetUpLine**, or directly with `\tikzset{line style/.style = ... }`.
Reminder about **line width**: There are a number of predefined styles that provide more "natural" ways of setting the line width. You can also redefine these styles.

predefined style	value of **line width**
ultra thin	0.1 pt
very thin	0.2 pt
thin	0.4 pt
semithick	0.6 pt
thick	0.8 pt
very thick	1.2 pt
ultra thick	1.6 pt

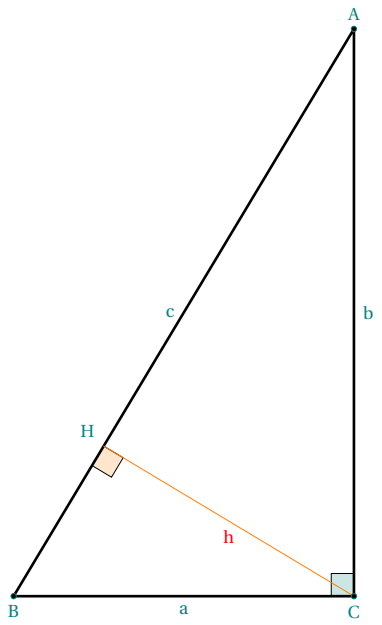
44.1. Use of \tkzSetUpLine

This macro defines the style of all the lines.

\tkzSetUpLine[⟨local options⟩]

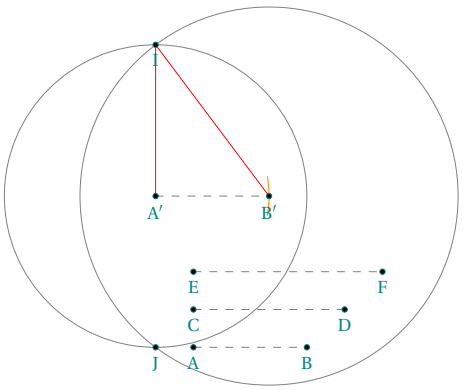
options	default	definition
color	black	color of the construction lines
line width	0.4pt	thickness of the construction lines
style	solid	style of construction lines
add	.2 and .2	changing the length of a line segment

44.1.1. Change line width



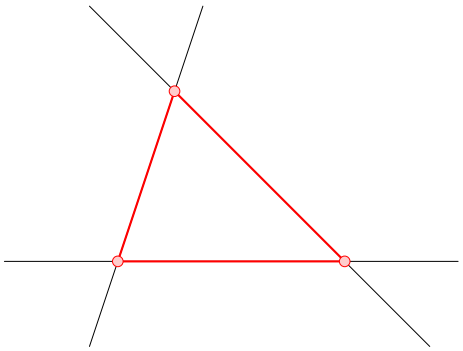

```
\begin{tikzpicture}[scale=.75]
\tkzSetUpLine[line width=1pt]
\begin{scope}[rotate=-90]
    \tkzDefPoints{0/6/A,10/0/B,10/6/C}
    \tkzDefPointBy[projection = onto B--A](C)
    \tkzGetPoint{H}
    \tkzMarkRightAngle[size=.4,
                       fill=teal!20](B,C,A)
    \tkzMarkRightAngle[size=.4,
                       fill=orange!20](B,H,C)
    \tkzDrawPolygon(A,B,C)
    \tkzDrawSegment[new](C,H)
\end{scope}
 \tkzLabelSegment[below](C,B){$a$}
 \tkzLabelSegment[right](A,C){$b$}
 \tkzLabelSegment[left](A,B){$c$}
 \tkzLabelSegment[color=red](C,H){$h$}
 \tkzDrawPoints(A,B,C)
 \tkzLabelPoints[above left](H)
 \tkzLabelPoints(B,C)
 \tkzLabelPoints[above](A)
\end{tikzpicture}
```

44.1.2. Change style of line




```
\begin{tikzpicture}[scale=.5]
\tikzset{line style/.style = {color = gray,
                              style=dashed}}
\tkzDefPoints{1/0/A,4/0/B,1/1/C,5/1/D}
\tkzDefPoints{1/2/E,6/2/F,0/4/A',3/4/B'}
\tkzCalcLength(C,D)
\tkzGetLength{rCD}
\tkzCalcLength(E,F)
\tkzGetLength{rEF}
\tkzInterCC[R](A',\rCD)(B',\rEF)
\tkzGetPoints{I}{J}
\tkzDrawLine(A',B')
\tkzCompass(A',B')
\tkzDrawSegments(A,B C,D E,F)
\tkzDefCircle[R](A',\rCD) \tkzGetPoint{a'}
\tkzDefCircle[R](B',\rEF)\tkzGetPoint{b'}
\tkzDrawCircles(A',a' B',b')
\begin{scope}
  \tkzSetUpLine[color=red]
  \tkzDrawSegments(A',I B',I)
\end{scope}
\tkzDrawPoints(A,B,C,D,E,F,A',B',I,J)
\tkzLabelPoints(A,B,C,D,E,F,A',B',I,J)
\end{tikzpicture}
```

44.1.3. Extending lines



```
\begin{tikzpicture}[scale=.75]
\tkzSetUpLine[add=.5 and .5]
\tkzDefPoints{0/0/A,4/0/B,1/3/C}
\tkzDrawLines(A,B B,C A,C)
\tkzDrawPolygon[red,thick](A,B,C)
\tkzSetUpPoint[size=4,circle,color=red,fill=red!20]
\tkzDrawPoints(A,B,C)
\end{tikzpicture}
```

45. Arc style

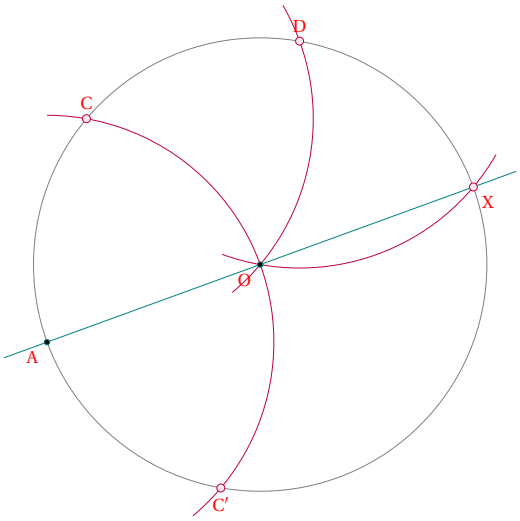
45.1. The macro \tkzSetUpArc

\tkzSetUpArc[⟨local options⟩]

options	default	definition
color	black	color of the lines
line width	0.4pt	thickness of the lines
style	solid	style of construction lines
delta	0	extends the arcs by **delta** degrees at each end

Since version 5.14, **delta** set with this macro only acts on the arcs drawn by **\tkzDrawArc**, **\tkzMarkArc** and **\tkzLabelArc**; the compass marks have their own **delta**, set with **\tkzSetUpCompass**.

45.1.1. Use of \tkzSetUpArc




```
\begin{tikzpicture}
\def\r{3} \def\angle{200}
\tkzSetUpArc[delta=10,color=purple,line width=.2pt]
\tkzSetUpLabel[font=\scriptsize,red]
\tkzDefPoint(0,0){O}
\tkzDefPoint(\angle:\r){A}
\tkzInterCC(O,A)(A,O) \tkzGetPoints{C'}{C}
\tkzInterCC(O,A)(C,O) \tkzGetPoints{D'}{D}
\tkzInterCC(O,A)(D,O) \tkzGetPoints{X'}{X}
\tkzDrawCircle(O,A)
\tkzDrawArc(A,C')(C)
\tkzDrawArc(C,O)(D)
\tkzDrawArc(D,O)(X)
\tkzDrawLine[add=.1 and .1](A,X)
\tkzDrawPoints(O,A)
\tkzSetUpPoint[size=3,color=purple,fill=purple!10]
\tkzDrawPoints(C,C',D,X)
\tkzLabelPoints[below left](O,A)
\tkzLabelPoints[below](C')
\tkzLabelPoints[below right](X)
\tkzLabelPoints[above](C,D)
\end{tikzpicture}
```

46. Compass style

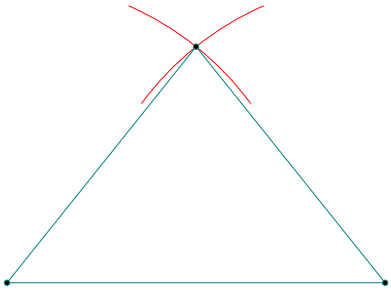
Compass marks help to understand the construction of a figure, by showing the arcs needed to obtain some points.

46.1. The macro \tkzSetUpCompass

\tkzSetUpCompass[⟨local options⟩]

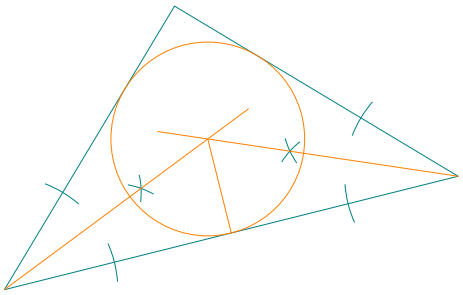
options	default	definition
color	black	color of the construction lines
line width	0.4pt	thickness of the construction lines
style	solid	style of lines: solid, dashed,dotted,...
delta	0	changes the length of the compass marks

46.1.1. Use of \tkzSetUpCompass



```
\begin{tikzpicture}
  \tkzSetUpCompass[color=red,delta=15]
  \tkzDefPoint(1,1){A}
  \tkzDefPoint(6,1){B}
  \tkzInterCC[R](A,4)(B,4) \tkzGetPoints{C}{D}
  \tkzCompass(A,C)
  \tkzCompass(B,C)
  \tkzDrawPolygon(A,B,C)
  \tkzDrawPoints(A,B,C)
\end{tikzpicture}
```

46.1.2. Use of \tkzSetUpCompass with \tkzShowLine



```
\begin{tikzpicture}[scale=.75]
\tkzSetUpStyle[bisector,size=2,gap=3]{showbi}
\tkzSetUpCompass[color=teal,line width=.3 pt]
\tkzDefPoints{0/1/A, 8/3/B, 3/6/C}
\tkzDrawPolygon(A,B,C)
\tkzDefLine[bisector](B,A,C) \tkzGetPoint{a}
\tkzDefLine[bisector](C,B,A) \tkzGetPoint{b}
\tkzShowLine[showbi](B,A,C)
\tkzShowLine[showbi](C,B,A)
\tkzInterLL(A,a)(B,b) \tkzGetPoint{I}
\tkzDefPointBy[projection= onto A--B](I)
\tkzGetPoint{H}
\tkzDrawCircle[new](I,H)
\tkzDrawSegments[new](I,H)
\tkzDrawLines[add=0 and .2,new](A,I B,I)
\end{tikzpicture}
```

47. Label style

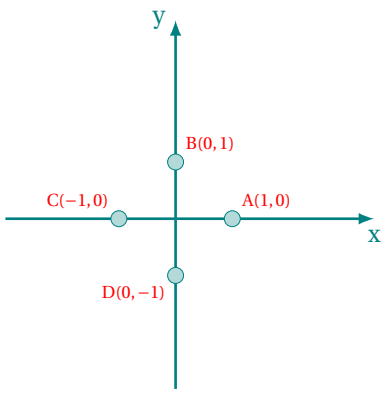
47.1. The macro \tkzSetUpLabel

The macro \tkzSetUpLabel defines the style of the point labels.

\tkzSetUpStyle[⟨local options⟩]

The options are the same as those of Ti*k*Z

47.1.1. Use of \tkzSetUpLabel




```
\begin{tikzpicture}[scale=.75]
  \tkzSetUpLabel[font=\scriptsize,red]
  \tkzSetUpStyle[line width=1pt,teal]{XY}
  \tkzInit[xmin=-3,xmax=3,ymin=-3,ymax=3]
  \tkzDrawX[noticks,XY]
  \tkzDrawY[noticks,XY]
  \tkzDefPoints{1/0/A,0/1/B,-1/0/C,0/-1/D}
  \tkzDrawPoints[teal,fill=teal!30,size=6](A,...,D)
  \tkzLabelPoint[above right](A){$A(1,0)$}
  \tkzLabelPoint[above right](B){$B(0,1)$}
  \tkzLabelPoint[above left](C){$C(-1,0)$}
  \tkzLabelPoint[below left](D){$D(0,-1)$}
\end{tikzpicture}
```

48. Own style

You can define your own style with **\tkzSetUpStyle**.

48.1. The macro \tkzSetUpStyle

\tkzSetUpStyle[⟨local options⟩]

The options are the same as those of Ti*k*Z

48.1.1. Use of \tkzSetUpStyle




```
\begin{tikzpicture}
  \tkzSetUpStyle[color=blue!20!black,fill=blue!20]{mystyle}
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(0,1){A}
  \tkzDrawPoints(O) % general style
  \tkzDrawPoints[mystyle,size=4](A) % my style
  \tkzLabelPoints(O,A)
\end{tikzpicture}
```

49. How to use arrows

In some countries, arrows are used to show that lines are parallel, or to represent rays, such as the sides of an angle.
Here are some examples of how to place these arrows. **tkz-euclide** loads the library **arrows.meta**.
\usetikzlibrary{arrows.meta}
This library provides different styles of arrow tips; the next examples use some of them.

49.1. Arrows at endpoints on segment, ray or line

Stealth, Triangle, To, Latex, ..., which can be combined with reversed. It is easy to place an arrow at one or both endpoints.

1. -Triangle and Segment


```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,4/0/B}
  \tkzDrawSegment[-Triangle](A,B)
\end{tikzpicture}
```

2. Stealth-Stealth and Segment


```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,4/0/B}
  \tkzDrawSegment[Stealth-Stealth](A,B)
\end{tikzpicture}
```

3. Latex-Latex and Line


```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,4/0/B}
  \tkzDrawLine[red,Latex-Latex](A,B)
  \tkzDrawPoints(A,B)
\end{tikzpicture}
```

4. To-To and Segment


```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,4/0/B}
  \tkzDrawSegment[To-To](A,B)
\end{tikzpicture}
```

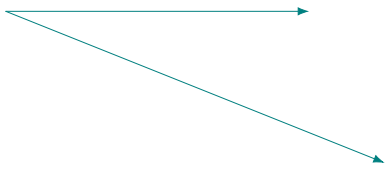
5. Latex-Latex and Segment


```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,4/0/B}
  \tkzDrawSegment[Latex-Latex](A,B)
\end{tikzpicture}
```

6. Latex- and Segment


```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,4/0/B}
  \tkzDrawSegment[Latex-](A,B)
\end{tikzpicture}
```

7. -Latex and Segments



```
\begin{tikzpicture}
 \tkzDefPoints{0/0/A,4/0/B,5/-2/C}
 \tkzDrawSegments[-Latex](A,B A,C)
\end{tikzpicture}
```

49.1.1. Scaling an arrow head



```
\begin{tikzpicture}
 \tkzDefPoints{0/0/A,4/0/B}
 \tkzDrawSegment[{Latex[scale=2]}-{Latex[scale=2]}](A,B)
\end{tikzpicture}
```

49.1.2. Using vector style

```
\tikzset{vector style/.style={>=Latex,->}}
```

You can redefine this style.


```
\begin{tikzpicture}
 \tkzDefPoints{0/0/A,4/0/B}
\tkzDrawSegment[vector style](A,B)
\end{tikzpicture}
```

49.2. Arrows on middle point of a line segment

Arrows on lines indicate that the lines are parallel. This depends on the country: in France we prefer to write outside the figure that (AB) ∥ (DC). The code is an adaptation of an answer by `Muzimuzhi Z` on the site tex.stackexchange.com.

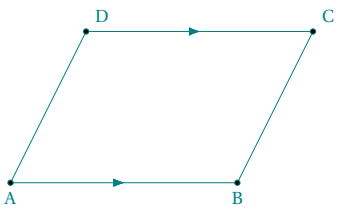
Syntax:

- `tkz arrow` (**Latex** by default)
- `tkz arrow=<arrow end tip>`
- `tkz arrow=<arrow end tip> at <pos> (<pos> = .5 by default)`
- `tkz arrow={<arrow end tip>[<arrow options>] at <pos>}` option possible **scale**

Example usages:

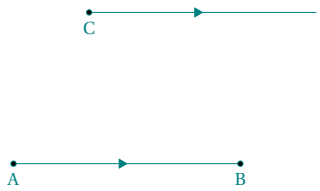
```
\tkzDrawSegment[tkz arrow=Stealth] (A,B)
\tkzDrawSegment[tkz arrow={To[scale=3] at .4}](A,B)
\tkzDrawSegment[tkz arrow={Latex[scale=5,blue] at .6}](A,B)
```

49.2.1. In a parallelogram




```
\begin{tikzpicture}
 \tkzDefPoints{0/0/A,3/0/B,4/2/C}
 \tkzDefParallelogram(A,B,C)
 \tkzGetPoint{D}
 \tkzDrawSegments[tkz arrow](A,B D,C)
 \tkzDrawSegments(B,C D,A)
 \tkzLabelPoints(A,B)
 \tkzLabelPoints[above right](C,D)
 \tkzDrawPoints(A,...,D)
\end{tikzpicture}
```

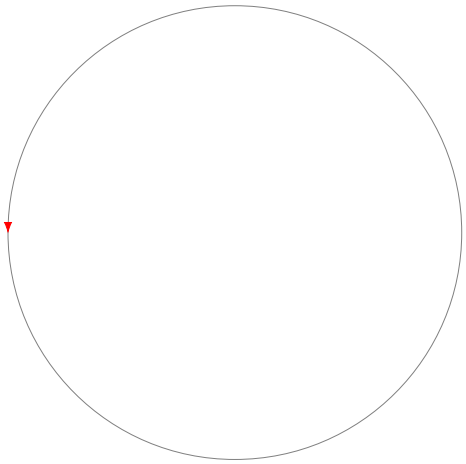
49.2.2. A line parallel to another one




```
\begin{tikzpicture}
 \tkzDefPoints{0/0/A,3/0/B,1/2/C}
 \tkzDefPointWith[colinear= at C](A,B)
 \tkzGetPoint{D}
 \tkzDrawSegments[tkz arrow=Triangle](A,B C,D)
 \tkzLabelPoints(A,B,C)
 \tkzDrawPoints(A,...,C)
\end{tikzpicture}
```

49.2.3. Arrow on a circle

You can place an arrow on the first quarter of a circle; a rotation moves the arrow.



```
\begin{tikzpicture}
\tkzDefPoints{0/0/A,3/0/B}
\begin{scope}[rotate=150]
 \tkzDrawCircle[tkz arrow={Latex[scale=2,red]}](A,B)
\end{scope}
\end{tikzpicture}
```

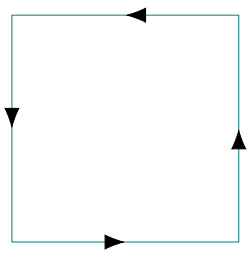
49.3. Arrows on all segments of a polygon

Some users asked me for a way to place an arrow on each side of a polygon. I used a style proposed by Paul Gaborit on the site tex.stackexchange.com.

```
\tikzset{tkz arrows/.style=
{postaction={on each path={tkz arrow={Latex[scale=2,color=black]}}}}}
```

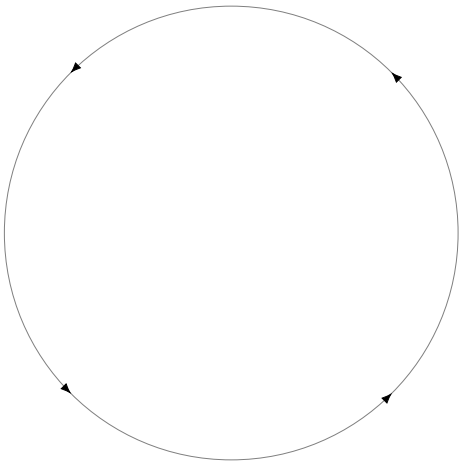
You can change this style. With **tkz arrows** you can place an arrow on each segment of a polygon.

49.3.1. Arrow on each segment with tkz arrows



```
\begin{tikzpicture}
 \tkzDefPoints{0/0/A,3/0/B}
 \tkzDefSquare(A,B) \tkzGetPoints{C}{D}
 \tkzDrawPolygon[tkz arrows](A,...,D)
\end{tikzpicture}
```

49.3.2. Using tkz arrows with a circle



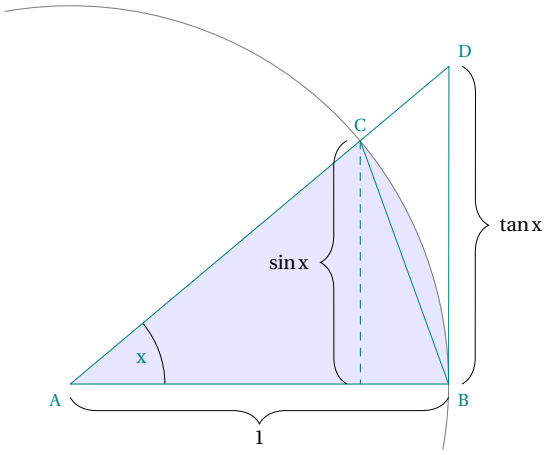
```
\begin{tikzpicture}
 \tkzDefPoints{0/0/A,3/0/B}
 \tkzDrawCircle[tkz arrows](A,B)
\end{tikzpicture}
```

Part X.

Examples

50. Different authors

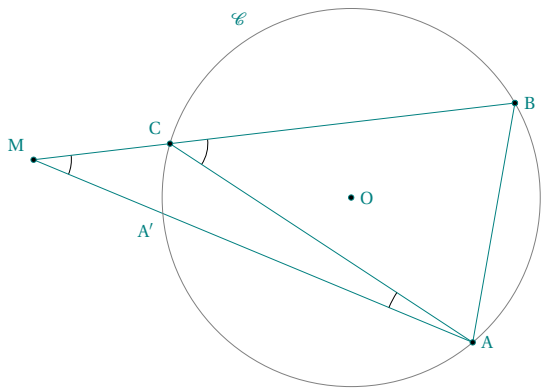
50.1. Code from Andrew Swan




```
\begin{tikzpicture}[scale=1.25]
\def\radius{4}
\def\angle{40}
\pgfmathsetmacro{\htan}{tan(\angle)}
\tkzDefPoint(0,0){A} \tkzDefPoint(0,\radius){F}
\tkzDefPoint(\radius,0){B}
\tkzDefPointBy[rotation= center A angle \angle](B)
\tkzGetPoint{C}
\tkzDefLine[perpendicular=through B,K=1](A,B)
\tkzGetPoint{b}
\tkzInterLL(A,C)(B,b) \tkzGetPoint{D}
\tkzDefLine[perpendicular=through C,K=-1](A,B)
\tkzGetPoint{c}
\tkzInterLL(C,c)(A,B) \tkzGetPoint{E}
\tkzDrawSector[fill=blue,opacity=0.1](A,B)(C)
\tkzDrawArc[thin](A,B)(F)
\tkzMarkAngle(B,A,C)
\tkzLabelAngle[pos=0.8](B,A,C){$x$}
\tkzDrawPolygon(A,B,D)
\tkzDrawSegments(C,B)
\tkzDrawSegments[dashed,thin](C,E)
\tkzLabelPoints[below left](A)
\tkzLabelPoints[below right](B)
\tkzLabelPoints[above](C)
\tkzLabelPoints[above right](D)
\begin{scope}[/pgf/decoration/raise=5pt]
\draw [decorate,decoration={brace,mirror,
   amplitude=10pt},xshift=0pt,yshift=-4pt]
(A) -- (B) node [black,midway,yshift=-20pt]
{\footnotesize $1$};
\draw [decorate,decoration={brace,amplitude=10pt},
       xshift=4pt,yshift=0pt]
(D) -- (B) node [black,midway,xshift=27pt]
{\footnotesize $\tan x$};
\draw [decorate,decoration={brace,amplitude=10pt},
       xshift=4pt,yshift=0pt]
(E) -- (C) node [black,midway,xshift=-27pt]
{\footnotesize $\sin x$};
\end{scope}
\end{tikzpicture}
```

50.2. Example by Dimitris Kapetas

In this example you need `mkpos=.2` with `\tkzMarkAngle`, because the angle $\widehat{CAM}$ is too small. Another possibility is to use `\tkzFillAngle`.

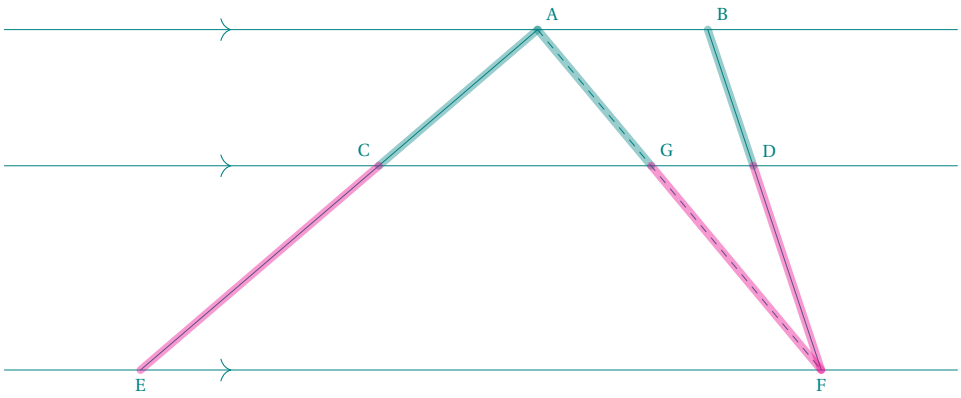



```
\begin{tikzpicture}[scale=1]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2.5,0){N}
  \tkzDefPoint(-4.2,0.5){M}
  \tkzDefPointBy[rotation=center O angle 30](N)
  \tkzGetPoint{B}
  \tkzDefPointBy[rotation=center O angle -50](N)
  \tkzGetPoint{A}
  \tkzInterLC[common=B](M,B)(O,B) \tkzGetFirstPoint{C}
  \tkzInterLC[common=A](M,A)(O,A) \tkzGetFirstPoint{A'}
  \tkzMarkAngle[mkpos=.2, size=0.5](A,C,B)
  \tkzMarkAngle[mkpos=.2, size=0.5](A,M,C)
  \tkzDrawSegments(A,C M,A M,B A,B)
  \tkzDrawCircle(O,N)
  \tkzLabelCircle[above left](O,N)(120){%
                $\mathcal{C}$}
  \begin{scope}[veclen]
  \tkzMarkAngle[mkpos=.2, size=1.2](C,A,M)
  \end{scope}
  \tkzDrawPoints(O, A, B, M, B, C)
  \tkzLabelPoints[right](O,A,B)
  \tkzLabelPoints[above left](M,C)
  \tkzLabelPoint[below left](A'){$A'$}
\end{tikzpicture}
```

50.3. Example by John Kitzmiller

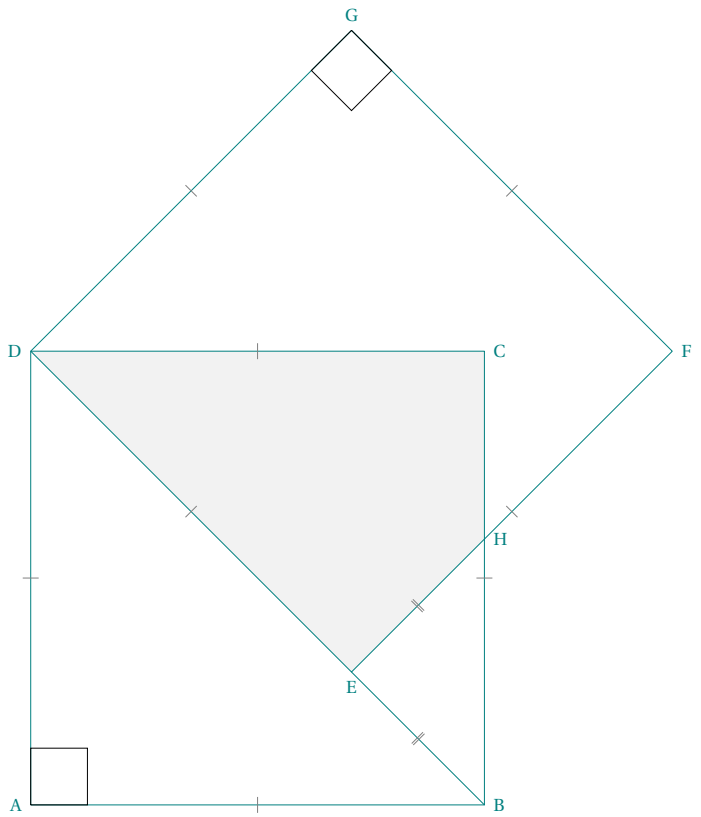
Prove that $\dfrac{AC}{CE} = \dfrac{BD}{DF}$.

Another interesting example from John, which shows how to use Ti*k*Z options such as **decoration** and **postaction** with **tkz-euclide**.



```
\begin{tikzpicture}[scale=1.5,decoration={markings,
  mark=at position 3cm with {\arrow[scale=2]{>}}}]
  \tkzDefPoints{0/0/E, 6/0/F, 0/1.8/P, 6/1.8/Q, 0/3/R, 6/3/S}
  \tkzDrawLines[postaction={decorate}](E,F P,Q R,S)
  \tkzDefPoints{3.5/3/A, 5/3/B}
  \tkzDrawSegments(E,A F,B)
  \tkzInterLL(E,A)(P,Q) \tkzGetPoint{C}
  \tkzInterLL(B,F)(P,Q) \tkzGetPoint{D}
  \tkzLabelPoints[above right](A,B)
  \tkzLabelPoints[below](E,F)
  \tkzLabelPoints[above left](C)
  \tkzDrawSegments[style=dashed](A,F)
  \tkzInterLL(A,F)(P,Q) \tkzGetPoint{G}
  \tkzLabelPoints[above right](D,G)
  \tkzDrawSegments[color=teal, line width=3pt, opacity=0.4](A,C A,G)
  \tkzDrawSegments[color=magenta, line width=3pt, opacity=0.4](C,E G,F)
  \tkzDrawSegments[color=teal, line width=3pt, opacity=0.4](B,D)
  \tkzDrawSegments[color=magenta, line width=3pt, opacity=0.4](D,F)
\end{tikzpicture}
```

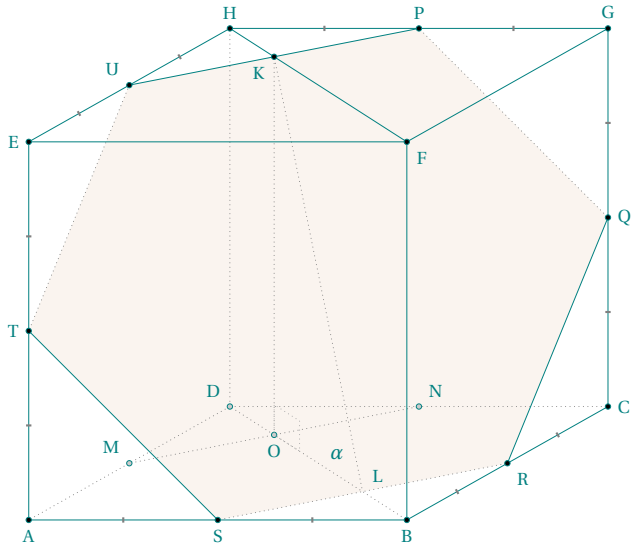
50.4. Example 1: from Indonesia



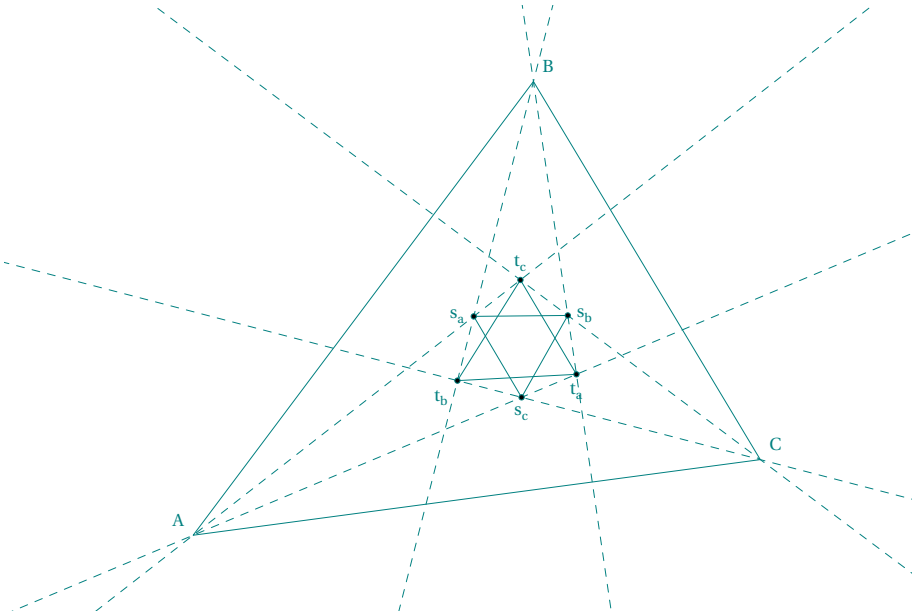
```
\begin{tikzpicture}[scale=3]
   \tkzDefPoints{0/0/A,2/0/B}
   \tkzDefSquare(A,B) \tkzGetPoints{C}{D}
   \tkzDefPointBy[rotation=center D angle 45](C)\tkzGetPoint{G}
   \tkzDefSquare(G,D)\tkzGetPoints{E}{F}
   \tkzInterLL(B,C)(E,F)\tkzGetPoint{H}
   \tkzFillPolygon[gray!10](D,E,H,C,D)
   \tkzDrawPolygon(A,...,D)\tkzDrawPolygon(D,...,G)
   \tkzDrawSegment(B,E)
   \tkzMarkSegments[mark=|,size=3pt,color=gray](A,B B,C C,D D,A E,F F,G G,D D,E)
   \tkzMarkSegments[mark=||,size=3pt,color=gray](B,E E,H)
   \tkzLabelPoints[left](A,D)
   \tkzLabelPoints[right](B,C,F,H)
   \tkzLabelPoints[above](G)\tkzLabelPoints[below](E)
   \tkzMarkRightAngles(D,A,B D,G,F)
\end{tikzpicture}
```

50.5. Example 2: from Indonesia

```
  \begin{tikzpicture}[pol/.style={fill=brown!40,opacity=.2},
      seg/.style={tkzdotted,color=gray}, hidden pt/.style={fill=gray!40},
       mra/.style={color=gray!70,tkzdotted,/tkzrightangle/size=.2},scale=2]
  \tkzDefPoints{0/0/A,2.5/0/B,1.33/0.75/D,0/2.5/E,2.5/2.5/F}
  \tkzDefLine[parallel=through D](A,B)  \tkzGetPoint{I1}
  \tkzDefLine[parallel=through B](A,D)  \tkzGetPoint{I2}
  \tkzInterLL(D,I1)(B,I2)               \tkzGetPoint{C}
  \tkzDefLine[parallel=through E](A,D)  \tkzGetPoint{I3}
  \tkzDefLine[parallel=through D](A,E)  \tkzGetPoint{I4}
  \tkzInterLL(E,I3)(D,I4)               \tkzGetPoint{H}
  \tkzDefLine[parallel=through F](E,H)  \tkzGetPoint{I5}
  \tkzDefLine[parallel=through H](E,F)  \tkzGetPoint{I6}
  \tkzInterLL(F,I5)(H,I6)               \tkzGetPoint{G}
  \tkzDefMidPoint(G,H) \tkzGetPoint{P}  \tkzDefMidPoint(G,C) \tkzGetPoint{Q}
  \tkzDefMidPoint(B,C) \tkzGetPoint{R}  \tkzDefMidPoint(A,B) \tkzGetPoint{S}
  \tkzDefMidPoint(A,E) \tkzGetPoint{T}  \tkzDefMidPoint(E,H) \tkzGetPoint{U}
  \tkzDefMidPoint(A,D) \tkzGetPoint{M}  \tkzDefMidPoint(D,C) \tkzGetPoint{N}
  \tkzInterLL(B,D)(S,R)\tkzGetPoint{L} \tkzInterLL(H,F)(U,P) \tkzGetPoint{K}
  \tkzDefLine[parallel=through K](D,H)  \tkzGetPoint{I7}
  \tkzInterLL(K,I7)(B,D)                \tkzGetPoint{O}
  \tkzFillPolygon[pol](P,Q,R,S,T,U)
  \tkzDrawSegments[seg](K,O K,L P,Q R,S T,U C,D H,D A,D M,N B,D)
  \tkzDrawSegments(E,H B,C G,F G,H G,C Q,R S,T U,P H,F)
  \tkzDrawPolygon(A,B,F,E)
  \tkzDrawPoints(A,B,C,E,F,G,H,P,Q,R,S,T,U,K) \tkzDrawPoints[hidden pt](M,N,O,D)
  \tkzMarkRightAngle[mra](L,O,K)
  \tkzMarkSegments[mark=|,size=1pt,thick,color=gray](A,S B,S B,R C,R
                   Q,C Q,G G,P H,P E,U H,U E,T A,T)
  \tkzLabelAngle[pos=.3](K,L,O){$\alpha$}
  \tkzLabelPoints[below](O,A,S,B)    \tkzLabelPoints[above](H,P,G)
  \tkzLabelPoints[left](T,E)         \tkzLabelPoints[right](C,Q)
  \tkzLabelPoints[above left](U,D,M) \tkzLabelPoints[above right](L,N)
  \tkzLabelPoints[below right](F,R)  \tkzLabelPoints[below left](K)
\end{tikzpicture}
```


H
P
G
U
K
E
F
Q
T
D
N
C
M
O
α
L
R
A
S
B

50.6. Illustration of the Morley theorem by Nicolas François

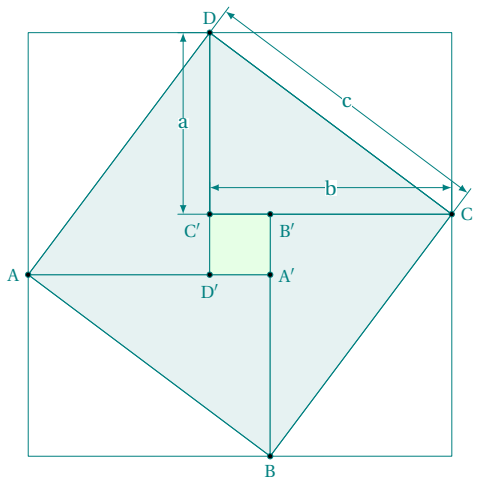



```
\begin{tikzpicture}
  \tkzInit[ymin=-3,ymax=5,xmin=-5,xmax=7]
  \tkzClip
  \tkzDefPoints{-2.5/-2/A,2/4/B,5/-1/C}
  \tkzFindAngle(C,A,B) \tkzGetAngle{anglea}
  \tkzDefPointBy[rotation=center A angle 1*\anglea/3](C) \tkzGetPoint{TA1}
  \tkzDefPointBy[rotation=center A angle 2*\anglea/3](C) \tkzGetPoint{TA2}
  \tkzFindAngle(A,B,C) \tkzGetAngle{angleb}
  \tkzDefPointBy[rotation=center B angle 1*\angleb/3](A) \tkzGetPoint{TB1}
  \tkzDefPointBy[rotation=center B angle 2*\angleb/3](A) \tkzGetPoint{TB2}
  \tkzFindAngle(B,C,A) \tkzGetAngle{anglec}
  \tkzDefPointBy[rotation=center C angle 1*\anglec/3](B) \tkzGetPoint{TC1}
  \tkzDefPointBy[rotation=center C angle 2*\anglec/3](B) \tkzGetPoint{TC2}
  \tkzInterLL(A,TA1)(B,TB2) \tkzGetPoint{U1}
  \tkzInterLL(A,TA2)(B,TB1) \tkzGetPoint{V1}
  \tkzInterLL(B,TB1)(C,TC2) \tkzGetPoint{U2}
  \tkzInterLL(B,TB2)(C,TC1) \tkzGetPoint{V2}
  \tkzInterLL(C,TC1)(A,TA2) \tkzGetPoint{U3}
  \tkzInterLL(C,TC2)(A,TA1) \tkzGetPoint{V3}
  \tkzDrawPolygons(A,B,C U1,U2,U3 V1,V2,V3)
  \tkzDrawLines[add=2 and 2,very thin,dashed](A,TA1 B,TB1 C,TC1 A,TA2 B,TB2 C,TC2)
  \tkzDrawPoints(U1,U2,U3,V1,V2,V3)
  \tkzLabelPoint[left](V1){$s_a$} \tkzLabelPoint[right](V2){$s_b$}
  \tkzLabelPoint[below](V3){$s_c$} \tkzLabelPoint[above left](A){$A$}
  \tkzLabelPoints[above right](B,C) \tkzLabelPoint(U1){$t_a$}
  \tkzLabelPoint[below left](U2){$t_b$} \tkzLabelPoint[above](U3){$t_c$}
\end{tikzpicture}
```

50.7. Gougu theorem (Pythagorean theorem) by Zhao Shuang

Gou gu theorem / Pythagorean Theorem by Zhao Shuang

Pythagoras was not the first person who discovered this theorem around the world. Ancient China discovered this theorem much earlier than him. So there is another name for the Pythagorean theorem in China, the Gou-Gu theorem. Zhao Shuang was an ancient Chinese mathematician. He rediscovered the "Gougu theorem", which is actually the Chinese version of the "Pythagorean theorem". Zhao Shuang used a method called the "cutting and compensation principle", he created a picture of "Pythagorean Round Square" Below the figure used to illustrate the proof of the "Gou gu theorem." (code from Nan Geng)

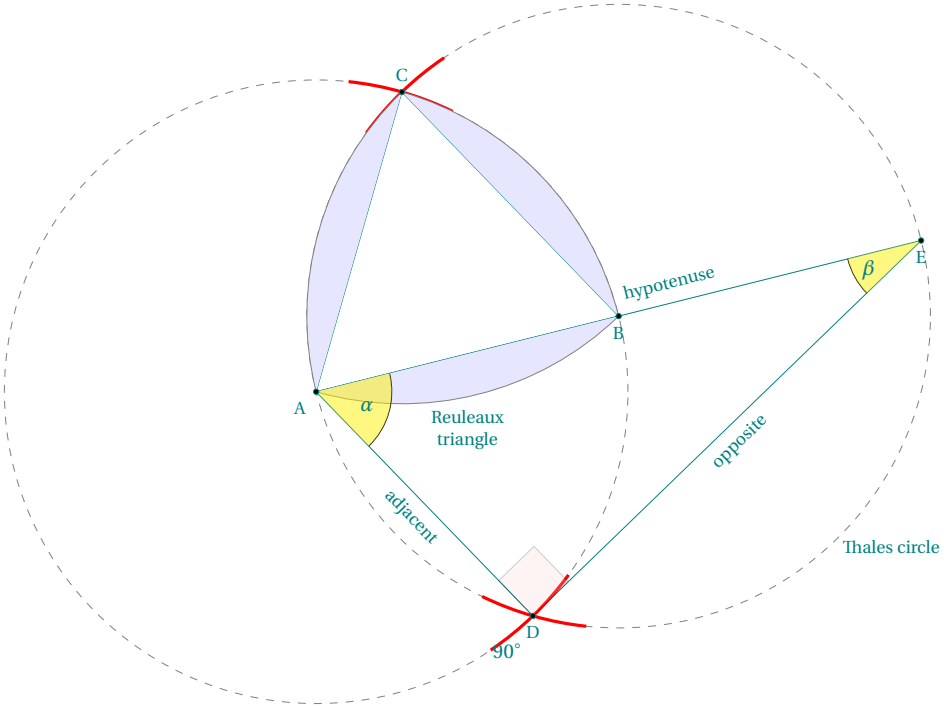



```
\begin{tikzpicture}[scale=.8]
  \tkzDefPoint(0,0){A} \tkzDefPoint(4,0){A'}
  \tkzInterCC[R](A, 5)(A', 3)
  \tkzGetSecondPoint{B}
  \tkzDefSquare(A,B)   \tkzGetPoints{C}{D}
  \tkzCalcLength(A,A') \tkzGetLength{lA}
  \tkzCalcLength(A',B) \tkzGetLength{lB}
  \pgfmathparse{\lA-\lB}
  \tkzInterLC[R](A,A')(A',\pgfmathresult)
  \tkzGetFirstPoint{D'}
  \tkzDefSquare(D',A')\tkzGetPoints{B'}{C'}
  \tkzDefLine[orthogonal=through D](D,D')
   \tkzGetPoint{d}
  \tkzDefLine[orthogonal=through A](A,A')
   \tkzGetPoint{a}
  \tkzDefLine[orthogonal=through C](C,C')
   \tkzGetPoint{c}
  \tkzInterLL(D,d)(C,c) \tkzGetPoint{E}
  \tkzInterLL(D,d)(A,a) \tkzGetPoint{F}
  \tkzDefSquare(E,F)\tkzGetPoints{G}{H}
  \tkzDrawPolygons[fill=teal!10](A,B,A' B,C,B'
     C,D,C' A,D',D)
  \tkzDrawPolygons(A,B,C,D E,F,G,H)
  \tkzDrawPolygon[fill=green!10](A',B',C',D')
  \tkzDrawSegment[dim={$a$,-10pt,}](D,C')
  \tkzDrawSegment[dim={$b$,-10pt,}](C,C')
  \tkzDrawSegment[dim={$c$,-10pt,}](C,D)
  \tkzDrawPoints[size=2](A,B,C,D,A',B',C',D')
  \tkzLabelPoints[left](A)
  \tkzLabelPoints[below](B)
  \tkzLabelPoints[right](C)
  \tkzLabelPoints[above](D)
  \tkzLabelPoints[right](A')
  \tkzLabelPoints[below right](B')
  \tkzLabelPoints[below left](C')
  \tkzLabelPoints[below](D')
\end{tikzpicture}
```

50.8. Reuleaux triangle

Reuleaux-triangle by Stefan Kottwitz

A well-known classic field of mathematics is geometry. You may know Euclidean geometry from school, with constructions by compass and ruler. Math teachers may be very interested in drawing geometry constructions and explanations. Underlying constructions can help us with general drawings where we would need intersections and tangents of lines and circles, even if it does not look like geometry. So, here, we will remember school geometry drawings. We will use the tkz-euclide package, which works on top of TikZ. We will construct an equilateral triangle. Then we extend it to get a Reuleaux triangle, and add annotations. The code is fully explained in the LaTeX Cookbook, Chapter 10, Advanced Mathematics, Drawing geometry pictures. Stefan Kottwitz



```
\begin{tikzpicture}
  \tkzDefPoint(0,0){A} \tkzDefPoint(4,1){B}
  \tkzInterCC(A,B)(B,A) \tkzGetPoints{C}{D}
  \tkzInterLC(A,B)(B,A) \tkzGetPoints{F}{E}
  \tkzDrawCircles[dashed](A,B B,A)
  \tkzDrawPolygons(A,B,C A,E,D)
  \tkzCompasss[color=red, very thick](A,C B,C A,D B,D)
  \begin{scope}
    \tkzSetUpArc[thick,delta=0]
    \tkzDrawArc[fill=blue!10](A,B)(C)
    \tkzDrawArc[fill=blue!10](B,C)(A)
    \tkzDrawArc[fill=blue!10](C,A)(B)
  \end{scope}
  \tkzMarkAngles(D,A,E A,E,D)
  \tkzFillAngles[fill=yellow,opacity=0.5](D,A,E A,E,D)
  \tkzMarkRightAngle[size=0.65,fill=red!20,opacity=0.2](A,D,E)
  \tkzLabelAngle[pos=0.7](D,A,E){$\alpha$}
  \tkzLabelAngle[pos=0.8](A,E,D){$\beta$}
  \tkzLabelAngle[pos=0.5,xshift=-1.4mm](A,D,D){$90^\circ$}
  \begin{scope}[font=\small]
    \tkzLabelSegment[below=0.6cm,align=center](A,B){Reuleaux\\triangle}
    \tkzLabelSegment[above right,sloped](A,E){hypotenuse}
    \tkzLabelSegment[below,sloped](D,E){opposite}
    \tkzLabelSegment[below,sloped](A,D){adjacent}
    \tkzLabelSegment[below right=4cm](A,E){Thales circle}
  \end{scope}
  \tkzLabelPoints[below left](A)
  \tkzLabelPoints(B,D)
  \tkzLabelPoint[above](C){$C$}
  \tkzLabelPoints(E)
  \tkzDrawPoints(A,...,E)

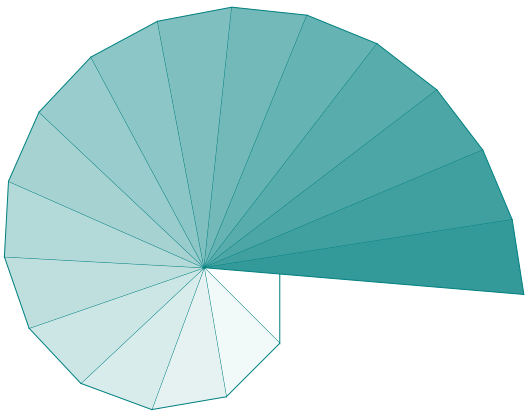
\end{tikzpicture}
```

51. Some interesting examples

51.1. Square root of the integers

Square root of the integers

How to get $1, \sqrt{2}, \sqrt{3}$ *with a rule and a compass.*

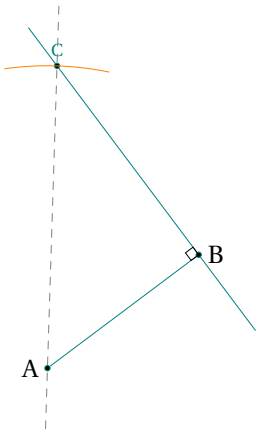


```
\begin{tikzpicture}
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(1,0){a0}
   \tkzDrawSegment(O,a0)
  \foreach \i [count=\j] in {0,...,16}{%
    \tkzDefPointWith[orthogonal normed](a\i,O)
    \tkzGetPoint{a\j}
       \pgfmathsetmacro{\c}{5*\i}
    \tkzDrawPolySeg[fill=teal!\c](a\i,a\j,O)}
 \end{tikzpicture}
```

51.2. About right triangles

About right triangle

We have a segment [AB] *and we want to determine a point* C *such that* AC = 8 *cm and* ABC *is a right triangle in* B.

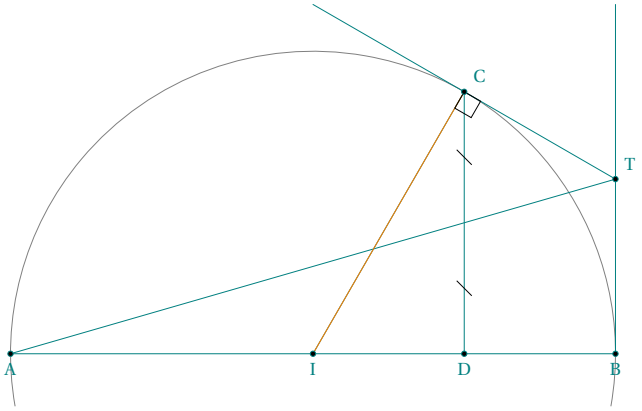



```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoint["$A$" left](2,1){A}
  \tkzDefPoint["$B$" right](6,4){B}
  \tkzDefPointWith[orthogonal,K=-1](B,A)
  \tkzDrawLine[add = .5 and .5](B,tkzPointResult)
  \tkzInterLC[R](B,tkzPointResult)(A,8)
  \tkzGetPoints{J}{C}
  \tkzDrawSegment(A,B)
  \tkzDrawPoints(A,B,C)
  \tkzCompass(A,C)
  \tkzMarkRightAngle(A,B,C)
  \tkzDrawLine[color=gray,style=dashed](A,C)
  \tkzLabelPoint[above](C){$C$}
\end{tikzpicture}
```

51.3. Archimedes

Archimedes

This is an ancient problem proved by the great Greek mathematician Archimedes . The figure below shows a semicircle, with diameter AB. *A tangent line is drawn and touches the semicircle at* B. *Another tangent line at a point,* C, *on the semicircle is drawn. We project the point* C *on the line segment* [AB] *on a point* D. *The two tangent lines intersect at the point* T. *Prove that the line* (AT) *bisects* (CD)

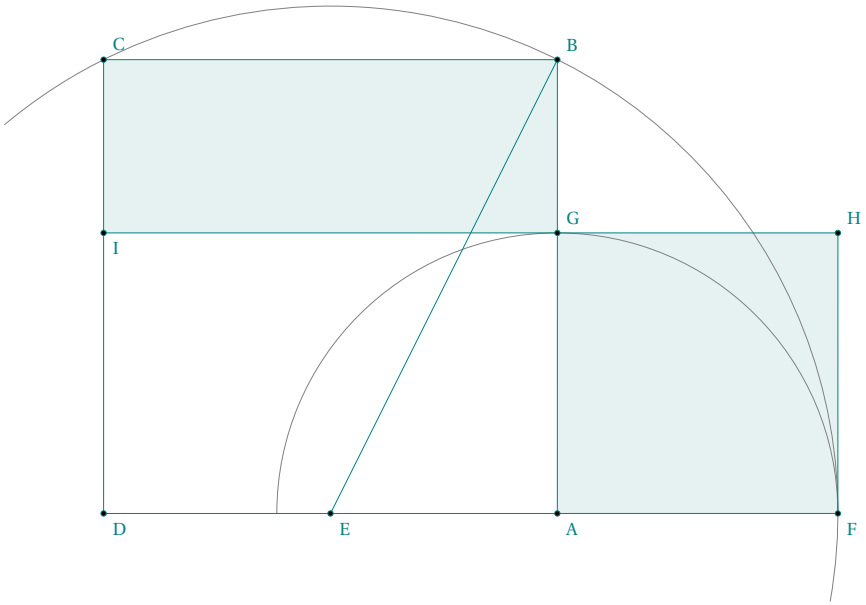


```
\begin{tikzpicture}[scale=1]
  \tkzDefPoint(0,0){A}\tkzDefPoint(6,0){D}
  \tkzDefPoint(8,0){B}\tkzDefPoint(4,0){I}
  \tkzDefLine[orthogonal=through D](A,D)
  \tkzInterLC[R](D,tkzPointResult)(I,4) \tkzGetSecondPoint{C}
  \tkzDefLine[orthogonal=through C](I,C)    \tkzGetPoint{c}
  \tkzDefLine[orthogonal=through B](A,B)    \tkzGetPoint{b}
  \tkzInterLL(C,c)(B,b) \tkzGetPoint{T}
  \tkzInterLL(A,T)(C,D) \tkzGetPoint{P}
  \tkzDrawArc(I,B)(A)
  \tkzDrawSegments(A,B A,T C,D I,C) \tkzDrawSegment[new](I,C)
  \tkzDrawLine[add = 1 and 0](C,T)   \tkzDrawLine[add = 0 and 1](B,T)
  \tkzMarkRightAngle(I,C,T)
  \tkzDrawPoints(A,B,I,D,C,T)
  \tkzLabelPoints(A,B,I,D)  \tkzLabelPoints[above right](C,T)
  \tkzMarkSegment[pos=.25,mark=s|](C,D) \tkzMarkSegment[pos=.75,mark=s|](C,D)
\end{tikzpicture}
```

51.3.1. Square and rectangle of same area; Golden section

Book II, proposition XI _Euclid's Elements_

To construct Square and rectangle of same area.

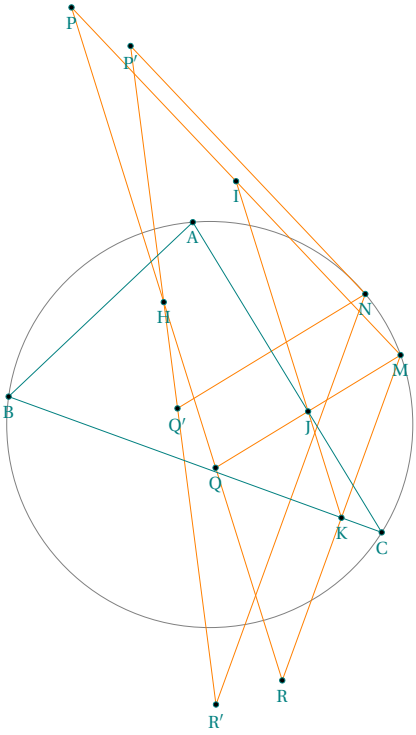



```
\begin{tikzpicture}[scale=.75]
 \tkzDefPoint(0,0){D} \tkzDefPoint(8,0){A}
 \tkzDefSquare(D,A) \tkzGetPoints{B}{C}
 \tkzDefMidPoint(D,A) \tkzGetPoint{E}
 \tkzInterLC(D,A)(E,B)\tkzGetSecondPoint{F}
 \tkzInterLC[near](B,A)(A,F)\tkzGetFirstPoint{G}
 \tkzDefSquare(A,F)\tkzGetFirstPoint{H}
 \tkzInterLL(C,D)(H,G)\tkzGetPoint{I}
 \tkzFillPolygon[teal!10](I,G,B,C)
 \tkzFillPolygon[teal!10](A,F,H,G)
 \tkzDrawArc[angles](E,B)(0,120)
 \tkzDrawSemiCircle(A,F)
 \tkzDrawSegments(A,F E,B H,I F,H)
 \tkzDrawPolygons(A,B,C,D)
 \tkzDrawPoints(A,...,I)
 \tkzLabelPoints[below right](A,E,D,F,I)
 \tkzLabelPoints[above right](C,B,G,H)
\end{tikzpicture}
```

51.3.2. Steiner Line and Simson Line

Steiner Line and Simson Line

Consider the triangle ABC and a point M on its circumcircle. The projections of M on the sides of the triangle are on a line (Steiner Line), The three closest points to M on lines AB, AC, and BC are collinear. It's the Simson Line.

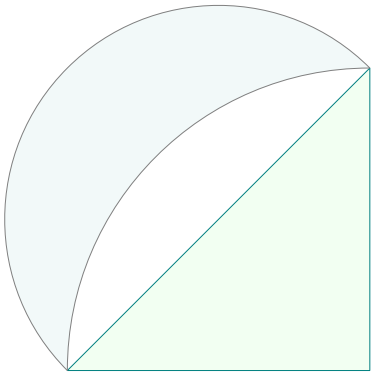



```
\begin{tikzpicture}[scale=.75,rotate=-20]
  \tkzDefPoint(0,0){B}
  \tkzDefPoint(2,4){A} \tkzDefPoint(7,0){C}
  \tkzDefCircle[circum](A,B,C)
  \tkzGetPoint{O}
  \tkzDrawCircle(O,A)
  \tkzCalcLength(O,A)
  \tkzGetLength{rOA}
  \tkzDefShiftPoint[O](40:\rOA){M}
  \tkzDefShiftPoint[O](60:\rOA){N}
  \tkzDefTriangleCenter[orthic](A,B,C)
  \tkzGetPoint{H}
  \tkzDefSpcTriangle[orthic,name=H](A,B,C){a,b,c}
  \tkzDefPointsBy[reflection=over A--B](M,N){P,P'}
  \tkzDefPointsBy[reflection=over A--C](M,N){Q,Q'}
  \tkzDefPointsBy[reflection=over C--B](M,N){R,R'}
  \tkzDefMidPoint(M,P)\tkzGetPoint{I}
  \tkzDefMidPoint(M,Q)\tkzGetPoint{J}
  \tkzDefMidPoint(M,R)\tkzGetPoint{K}
  \tkzDrawSegments[new](P,R M,P M,Q M,R N,P'
   N,Q' N,R' P',R' I,K)
  \tkzDrawPolygons(A,B,C)
  \tkzDrawPoints(A,B,C,H,M,N,P,Q,R,P',Q',R',I,J,K)
  \tkzLabelPoints(A,B,C,H,M,N,P,Q,R,P',Q',R',I,J,K)
\end{tikzpicture}
```

51.4. Lune of Hippocrates

Lune of Hippocrates

From wikipedia : In geometry, the lune of Hippocrates, named after Hippocrates of Chios, is a lune bounded by arcs of two circles, the smaller of which has as its diameter a chord spanning a right angle on the larger circle.In the first figure, the area of the lune is equal to the area of the triangle ABC. Hippocrates of Chios (ancient Greek mathematician,)

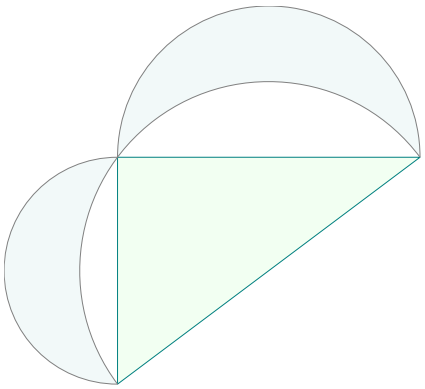


```
\begin{tikzpicture}
 \tkzInit[xmin=-2,xmax=5,ymin=-1,ymax=6]
 \tkzClip % allows you to define a bounding box
   % large enough
  \tkzDefPoint(0,0){A}\tkzDefPoint(4,0){B}
  \tkzDefSquare(A,B)
  \tkzGetFirstPoint{C}
  \tkzDrawPolygon[fill=green!5](A,B,C)
   \begin{scope}
     \tkzClipCircle[out](B,A)
     \tkzDefMidPoint(C,A) \tkzGetPoint{M}
     \tkzDrawSemiCircle[fill=teal!5](M,C)
   \end{scope}
   \tkzDrawArc[delta=0](B,C)(A)
\end{tikzpicture}
```

51.5. Lunes of Hasan Ibn al-Haytham

Lune of Hippocrates

From wikipedia : the Arab mathematician Hasan Ibn al-Haytham (Latinized name Alhazen) showed that two lunes, formed on the two sides of a right triangle, whose outer boundaries are semicircles and whose inner boundaries are formed by the circumcircle of the triangle, then the areas of these two lunes added together are equal to the area of the triangle. The lunes formed in this way from a right triangle are known as the lunes of Alhazen.

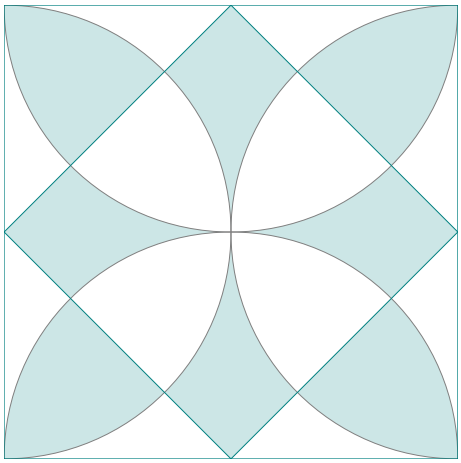


```
\begin{tikzpicture}[scale=.5,rotate=180]
  \tkzInit[xmin=-1,xmax=11,ymin=-4,ymax=7]
  \tkzClip
  \tkzDefPoints{0/0/A,8/0/B}
  \tkzDefTriangle[pythagore,swap](A,B)
  \tkzGetPoint{C}
  \tkzDrawPolygon[fill=green!5](A,B,C)
  \tkzDefMidPoint(C,A) \tkzGetPoint{I}
  \begin{scope}
    \tkzClipCircle[out](I,A)
    \tkzDefMidPoint(B,A) \tkzGetPoint{x}
    \tkzDrawSemiCircle[fill=teal!5](x,A)
    \tkzDefMidPoint(B,C) \tkzGetPoint{y}
    \tkzDrawSemiCircle[fill=teal!5](y,B)
  \end{scope}
  \tkzSetUpCompass[/tkzcompass/delta=0]
      \tkzDefMidPoint(C,A) \tkzGetPoint{z}
  \tkzDrawSemiCircle(z,A)
\end{tikzpicture}
```

51.6. About clipping circles

About clipping circles

The problem is the management of the bounding box. First you have to define a rectangle in which the figure will be inserted. This is done with the first two lines.



```
\begin{tikzpicture}
  \tkzInit[xmin=0,xmax=6,ymin=0,ymax=6]
  \tkzClip
  \tkzDefPoints{0/0/A, 6/0/B}
  \tkzDefSquare(A,B)       \tkzGetPoints{C}{D}
  \tkzDefMidPoint(A,B)         \tkzGetPoint{M}
  \tkzDefMidPoint(A,D)         \tkzGetPoint{N}
  \tkzDefMidPoint(B,C)         \tkzGetPoint{O}
  \tkzDefMidPoint(C,D)         \tkzGetPoint{P}
 \begin{scope}
  \tkzClipCircle[out](M,B) \tkzClipCircle[out](P,D)
  \tkzFillPolygon[teal!20](M,N,P,O)
 \end{scope}
 \begin{scope}
   \tkzClipCircle[out](N,A) \tkzClipCircle[out](O,C)
   \tkzFillPolygon[teal!20](M,N,P,O)
 \end{scope}
 \begin{scope}
   \tkzClipCircle(P,C) \tkzClipCircle(N,A)
   \tkzFillPolygon[teal!20](N,P,D)
 \end{scope}
 \begin{scope}
     \tkzClipCircle(O,C) \tkzClipCircle(P,C)
     \tkzFillPolygon[teal!20](P,C,O)
 \end{scope}
 \begin{scope}
     \tkzClipCircle(M,B)  \tkzClipCircle(O,B)
     \tkzFillPolygon[teal!20](O,B,M)
 \end{scope}
 \begin{scope}
     \tkzClipCircle(N,A) \tkzClipCircle(M,A)
     \tkzFillPolygon[teal!20](A,M,N)
 \end{scope}
 \tkzDrawSemiCircles(M,B N,A O,C P,D)
 \tkzDrawPolygons(A,B,C,D M,N,P,O)
 \end{tikzpicture}
```

51.7. Similar isosceles triangles

Similar isosceles triangles

The following is from the excellent site ***Descartes et les Mathématiques****. I did not modify the text and I am only the author of the programming of the figures.* `http://debart.pagesperso-orange.fr/seconde/triangle.html`

The following is from the excellent site **Descartes et les Mathématiques**. I did not modify the text; I only programmed the figures.
`http://debart.pagesperso-orange.fr/seconde/triangle.html`
Bibliography:

- Géométrie au Bac - Tangente, special issue no. 8 - Exercise 11, page 11
- Elisabeth Busser and Gilles Cohen: 200 nouveaux problèmes du "Monde" - POLE 2007 (200 new problems of "Le Monde")
- Affaire de logique n° 364 - Le Monde February 17, 2004

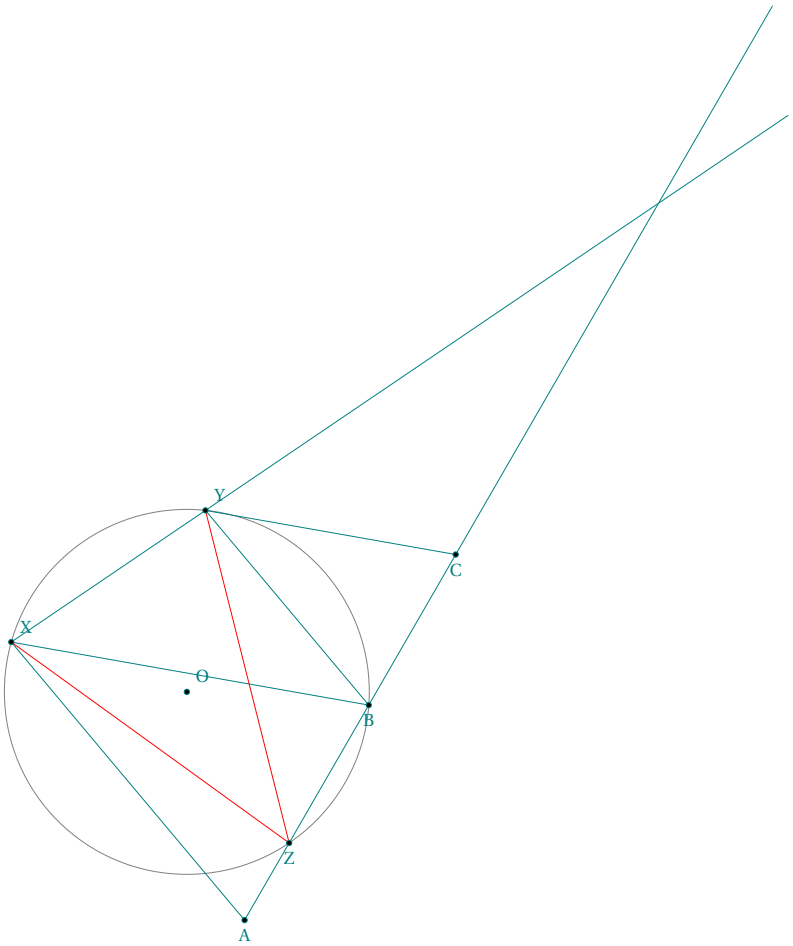
Two statements were proposed, one by the magazine *Tangente* and the other by *Le Monde*.

Editor of the magazine "Tangente": Two similar isosceles triangles AXB and BYC are constructed with main vertices X and Y, such that A, B and C are aligned and that these triangles are "indirect". Let α be the vertex angle $\widehat{AXB} = \widehat{BYC}$. We then construct a third isosceles triangle XZY, similar to the first two, with main vertex Z and "indirect". Prove that the point Z lies on the straight line (AC).

Editor of "Le Monde": We construct two similar isosceles triangles AXB and BYC with principal vertices X and Y, such that A, B and C are aligned and that these triangles are "indirect". Let α be the vertex angle $\widehat{AXB} = \widehat{BYC}$. The point Z of the segment [AC] is equidistant from the two vertices X and Y.
At what angle does it see these two vertices?

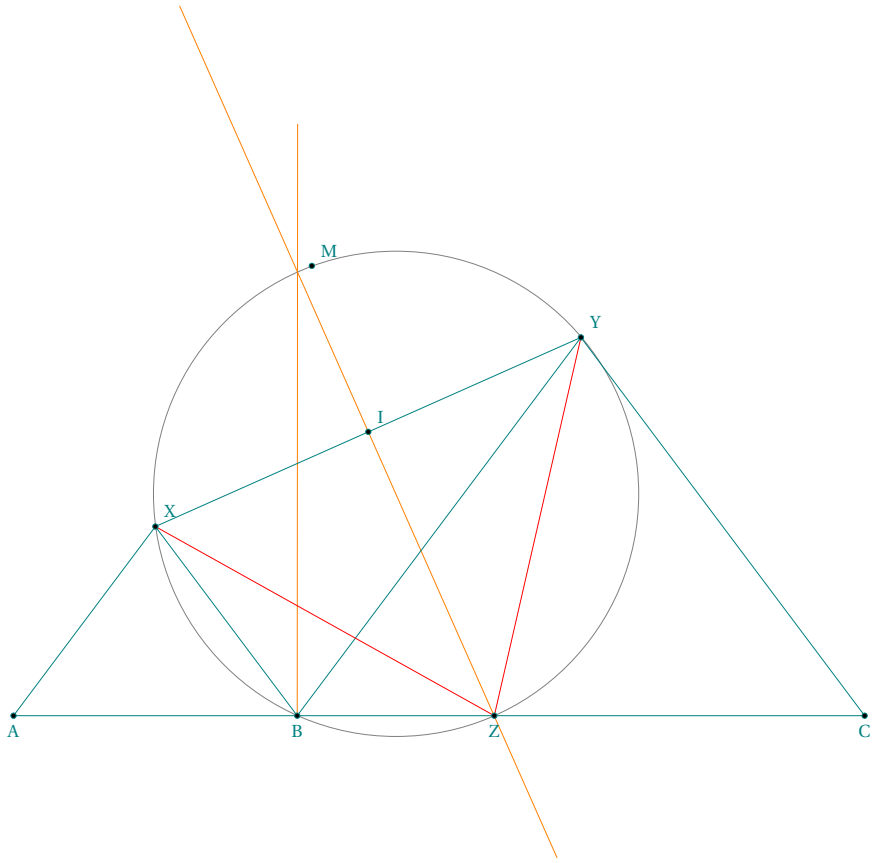
The constructions and their codes are on the next two pages, but you can search before looking. In both cases, the programming follows my reasoning (I think...).

51.8. Revised version of "Tangente"




```
\begin{tikzpicture}[scale=.8,rotate=60]
  \tkzDefPoint(6,0){X}   \tkzDefPoint(3,3){Y}
  \tkzDefShiftPoint[X](-110:6){A}    \tkzDefShiftPoint[X](-70:6){B}
  \tkzDefShiftPoint[Y](-110:4.2){A'} \tkzDefShiftPoint[Y](-70:4.2){B'}
  \tkzDefPointBy[translation= from A' to B ](Y) \tkzGetPoint{Y}
  \tkzDefPointBy[translation= from A' to B ](B') \tkzGetPoint{C}
  \tkzInterLL(A,B)(X,Y) \tkzGetPoint{O}
  \tkzDefMidPoint(X,Y) \tkzGetPoint{I}
  \tkzDefPointWith[orthogonal](I,Y)
  \tkzInterLL(I,tkzPointResult)(A,B) \tkzGetPoint{Z}
  \tkzDefCircle[circum](X,Y,B) \tkzGetPoint{O}
  \tkzDrawCircle(O,X)
  \tkzDrawLines[add = 0 and 1.5](A,C) \tkzDrawLines[add = 0 and 3](X,Y)
  \tkzDrawSegments(A,X B,X B,Y C,Y)   \tkzDrawSegments[color=red](X,Z Y,Z)
  \tkzDrawPoints(A,B,C,X,Y,O,Z)
  \tkzLabelPoints(A,B,C,Z)   \tkzLabelPoints[above right](X,Y,O)
\end{tikzpicture}
```

51.9. "Le Monde" version

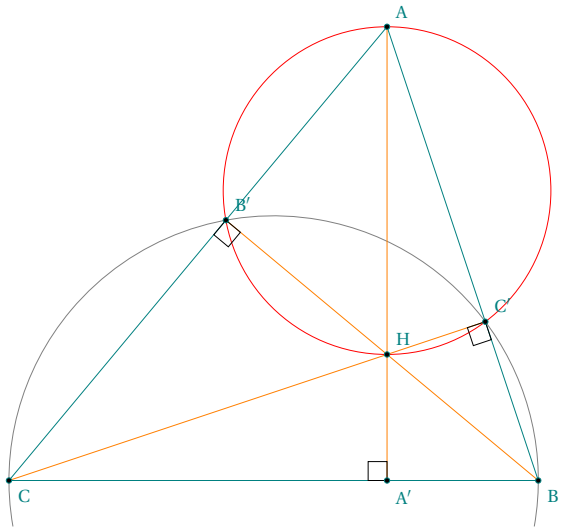



```
\begin{tikzpicture}[scale=1.25]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(3,0){B}
  \tkzDefPoint(9,0){C}
  \tkzDefPoint(1.5,2){X}
  \tkzDefPoint(6,4){Y}
  \tkzDefCircle[circum](X,Y,B) \tkzGetPoint{O}
  \tkzDefMidPoint(X,Y)                 \tkzGetPoint{I}
  \tkzDefPointWith[orthogonal](I,Y)    \tkzGetPoint{i}
  \tkzDrawLines[add = 2 and 1,color=orange](I,i)
  \tkzInterLL(I,i)(A,B)                \tkzGetPoint{Z}
  \tkzInterLC(I,i)(O,B)                \tkzGetFirstPoint{M}
  \tkzDefPointWith[orthogonal](B,Z)    \tkzGetPoint{b}
  \tkzDrawCircle(O,B)
  \tkzDrawLines[add = 0 and 2,color=orange](B,b)
  \tkzDrawSegments(A,X B,X B,Y C,Y A,C X,Y)
  \tkzDrawSegments[color=red](X,Z Y,Z)
  \tkzDrawPoints(A,B,C,X,Y,Z,M,I)
  \tkzLabelPoints(A,B,C,Z)
  \tkzLabelPoints[above right](X,Y,M,I)
\end{tikzpicture}
```

51.10. Triangle altitudes

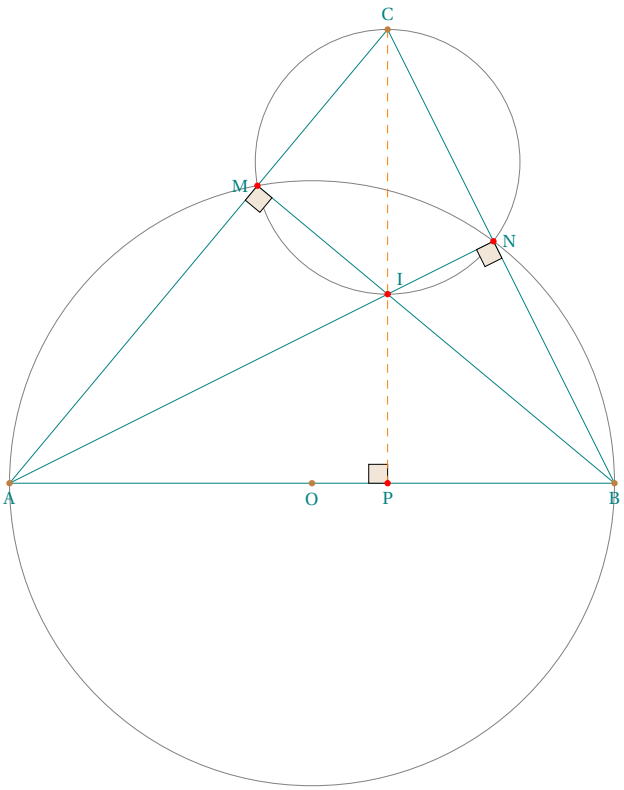
Triangle altitudes

From Wikipedia : The following is again from the excellent site ***Descartes et les Mathématiques*** *(Descartes and the Mathematics).* `http://debart.pagesperso-orange.fr/geoplan/geometrie_triangle.html`. *The three altitudes of a triangle intersect at the same H-point.*




```
\begin{tikzpicture}
   \tkzDefPoint(0,0){C} \tkzDefPoint(7,0){B}
   \tkzDefPoint(5,6){A}
   \tkzDefMidPoint(C,B) \tkzGetPoint{I}
   \tkzInterLC(A,C)(I,B)
   \tkzGetFirstPoint{B'}
   \tkzInterLC(A,B)(I,B)
   \tkzGetSecondPoint{C'}
   \tkzInterLL(B,B')(C,C') \tkzGetPoint{H}
   \tkzInterLL(A,H)(C,B) \tkzGetPoint{A'}
   \tkzDefCircle[circum](A,B',C') \tkzGetPoint{O}
   \tkzDrawArc(I,B)(C)
   \tkzDrawPolygon(A,B,C)
   \tkzDrawCircle[color=red](O,A)
   \tkzDrawSegments[color=orange](B,B' C,C' A,A')
   \tkzMarkRightAngles(C,B',B B,C',C C,A',A)
   \tkzDrawPoints(A,B,C,A',B',C',H)
   \tkzLabelPoints[above right](A,B',C',H)
   \tkzLabelPoints[below right](B,C,A')
\end{tikzpicture}
```

51.11. Altitudes - other construction

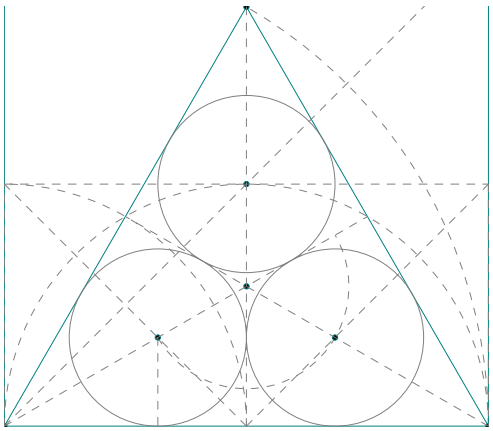



```
\begin{tikzpicture}
\tkzDefPoint(0,0){A} \tkzDefPoint(8,0){B}
\tkzDefPoint(5,6){C}
\tkzDefMidPoint(A,B)\tkzGetPoint{O}
\tkzDefPointBy[projection=onto A--B](C) \tkzGetPoint{P}
\tkzInterLC[common=A](C,A)(O,A)
\tkzGetFirstPoint{M}
\tkzInterLC(C,B)(O,A)
\tkzGetSecondPoint{N}
\tkzInterLL(B,M)(A,N)\tkzGetPoint{I}
\tkzDefCircle[diameter](A,B)\tkzGetPoint{x}
\tkzDefCircle[diameter](I,C)\tkzGetPoint{y}
\tkzDrawCircles(x,A y,C)
\tkzDrawSegments(C,A C,B A,B B,M A,N)
\tkzMarkRightAngles[fill=brown!20](A,M,B A,N,B A,P,C)
\tkzDrawSegment[style=dashed,color=orange](C,P)
\tkzLabelPoints(O,A,B,P)
\tkzLabelPoint[left](M){$M$}
\tkzLabelPoint[right](N){$N$}
\tkzLabelPoint[above](C){$C$}
\tkzLabelPoint[above right](I){$I$}
\tkzDrawPoints[color=red](M,N,P,I)
\tkzDrawPoints[color=brown](O,A,B,C)
\end{tikzpicture}
```

51.12. Three circles in an equilateral triangle

Three circles in an Equilateral Triangle

From Wikipedia : In geometry, the Malfatti circles are three circles inside a given triangle such that each circle is tangent to the other two and to two sides of the triangle. They are named after Gian Francesco Malfatti, who made early studies of the problem of constructing these circles in the mistaken belief that they would have the largest possible total area of any three disjoint circles within the triangle. Below is a study of a particular case with an equilateral triangle and three identical circles.

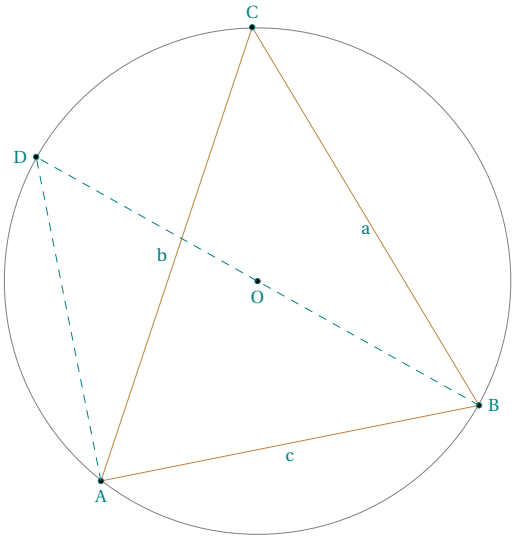


```
\begin{tikzpicture}[scale=.8]
  \tkzDefPoints{0/0/A,8/0/B,0/4/a,8/4/b,8/8/c}
  \tkzDefTriangle[equilateral](A,B) \tkzGetPoint{C}
  \tkzDefMidPoint(A,B) \tkzGetPoint{M}
  \tkzDefMidPoint(B,C) \tkzGetPoint{N}
  \tkzDefMidPoint(A,C) \tkzGetPoint{P}
  \tkzInterLL(A,N)(M,a) \tkzGetPoint{Ia}
  \tkzDefPointBy[projection = onto A--B](Ia)
  \tkzGetPoint{ha}
  \tkzInterLL(B,P)(M,b) \tkzGetPoint{Ib}
  \tkzDefPointBy[projection = onto A--B](Ib)
  \tkzGetPoint{hb}
  \tkzInterLL(A,c)(M,C) \tkzGetPoint{Ic}
  \tkzDefPointBy[projection = onto A--C](Ic)
  \tkzGetPoint{hc}
  \tkzInterLL(A,Ia)(B,Ib) \tkzGetPoint{G}
  \tkzDefSquare(A,B) \tkzGetPoints{D}{E}
  \tkzDrawPolygon(A,B,C)
  \tkzClipBB
  \tkzDrawSemiCircles[gray,dashed](M,B A,M
  A,B B,A G,Ia)
  \tkzDrawCircles[gray](Ia,ha Ib,hb Ic,hc)
  \tkzDrawPolySeg(A,E,D,B)
  \tkzDrawPoints(A,B,C,G,Ia,Ib,Ic)
  \tkzDrawSegments[gray,dashed](C,M A,N B,P
   M,a M,b A,a a,b b,B A,D Ia,ha)
\end{tikzpicture}
```

51.13. Law of sines

Law of sines

From wikipedia : *In trigonometry, the law of sines, sine law, sine formula, or sine rule is an equation relating the lengths of the sides of a triangle (any shape) to the sines of its angles.*




```
\begin{tikzpicture}
\tkzDefPoints{0/0/A,5/1/B,2/6/C}
\tkzDefTriangleCenter[circum](A,B,C)
 \tkzGetPoint{O}
\tkzDefPointBy[symmetry= center O](B)
 \tkzGetPoint{D}
\tkzDrawPolygon[color=brown](A,B,C)
\tkzDrawCircle(O,A)
\tkzDrawPoints(A,B,C,D,O)
\tkzDrawSegments[dashed](B,D A,D)
\tkzLabelPoint[left](D){$D$}
\tkzLabelPoint[below](A){$A$}
\tkzLabelPoint[above](C){$C$}
\tkzLabelPoint[right](B){$B$}
\tkzLabelPoint[below](O){$O$}
\tkzLabelSegment(B,C){$a$}
\tkzLabelSegment[left](A,C){$b$}
\tkzLabelSegment(A,B){$c$}
\end{tikzpicture}
```

In the triangle ABC:

$$\frac{a}{\sin A} = \frac{b}{\sin B} = \frac{c}{\sin C} \tag{1}$$

$$\widehat{C} = \widehat{D}$$

$$\frac{c}{2R} = \sin D = \sin C \tag{2}$$

Then

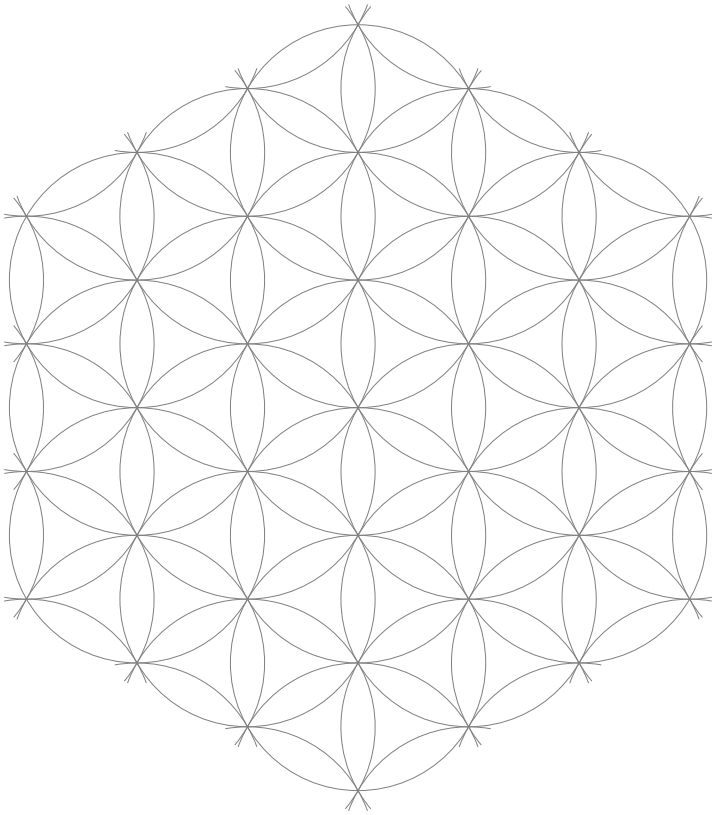
$$\frac{c}{\sin C} = 2R$$

51.14. Flower of Life

Book IV, proposition XI _Euclid's Elements_

Sacred geometry can be described as a belief system attributing a religious or cultural value to many of the fundamental forms of space and time. According to this belief system, the basic patterns of existence are perceived as sacred because in contemplating them one is contemplating the origin of all things. By studying the nature of these forms and their relationship to each other, one may seek to gain insight into the scientific, philosophical, psychological, aesthetic and mystical laws of the universe. The Flower of Life is considered to be a symbol of sacred geometry, said to contain ancient, religious value depicting the fundamental forms of space and time. In this sense, it is a visual expression of the connections life weaves through all mankind, believed by some to contain a type of Akashic Record of basic information of all living things.

One of the beautiful arrangements of circles found at the Temple of Osiris at Abydos, Egypt (Rawles 1997). Weisstein, Eric W. "Flower of Life." From MathWorld–A Wolfram Web Resource.
http://mathworld.wolfram.com/FlowerofLife.html

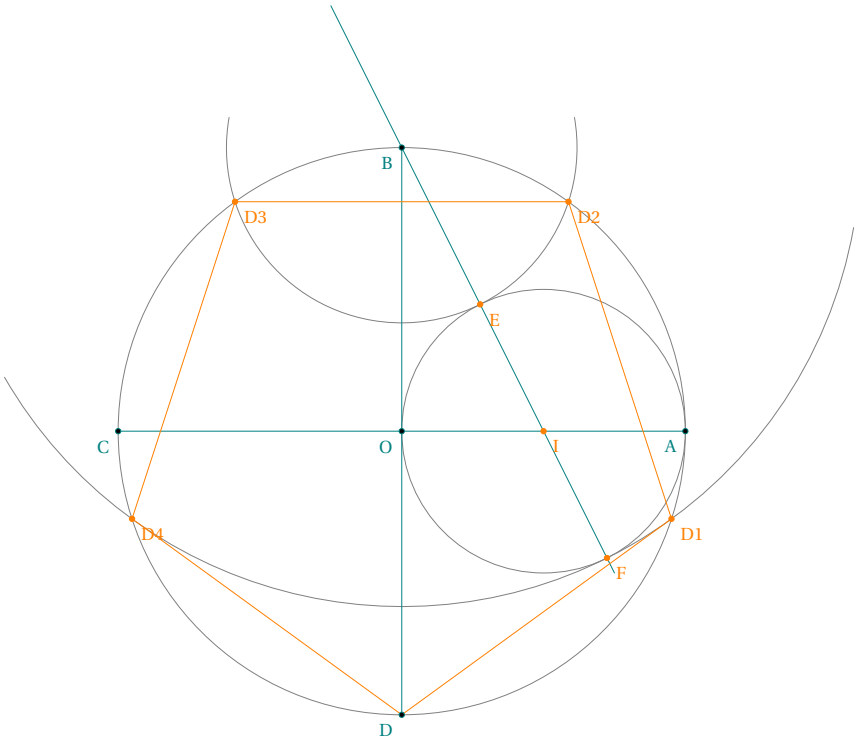


```
\begin{tikzpicture}[scale=.75]
  \tkzSetUpLine[line width=2pt,color=teal!80!black]
  \tkzSetUpCompass[line width=2pt,color=teal!80!black]
   \tkzDefPoint(0,0){O}  \tkzDefPoint(2.25,0){A}
   \tkzDrawCircle(O,A)
\foreach \i in {0,...,5}{
   \tkzDefPointBy[rotation= center O angle 30+60*\i](A)\tkzGetPoint{a\i}
   \tkzDefPointBy[rotation= center {a\i} angle  120](O)\tkzGetPoint{b\i}
   \tkzDefPointBy[rotation= center {a\i} angle  180](O)\tkzGetPoint{c\i}
   \tkzDefPointBy[rotation= center {c\i} angle  120](a\i)\tkzGetPoint{d\i}
   \tkzDefPointBy[rotation= center {c\i} angle   60](d\i)\tkzGetPoint{f\i}
   \tkzDefPointBy[rotation= center {d\i} angle   60](b\i)\tkzGetPoint{e\i}
   \tkzDefPointBy[rotation= center {f\i} angle   60](d\i)\tkzGetPoint{g\i}
   \tkzDefPointBy[rotation= center {d\i} angle   60](e\i)\tkzGetPoint{h\i}
   \tkzDefPointBy[rotation= center {e\i} angle  180](b\i)\tkzGetPoint{k\i}
   \tkzDrawCircle(a\i,O)
   \tkzDrawCircle(b\i,a\i)
   \tkzDrawCircle(c\i,a\i)
   \tkzDrawArc[rotate](f\i,d\i)(-120)
   \tkzDrawArc[rotate](e\i,d\i)(180)
   \tkzDrawArc[rotate](d\i,f\i)(180)
   \tkzDrawArc[rotate](g\i,f\i)(60)
   \tkzDrawArc[rotate](h\i,d\i)(60)
   \tkzDrawArc[rotate](k\i,e\i)(60)
}
   \tkzClipCircle(O,f0)
\end{tikzpicture}
```

51.15. Pentagon in a circle

Book IV, proposition XI _Euclid's Elements_

To inscribe an equilateral and equiangular pentagon in a given circle.

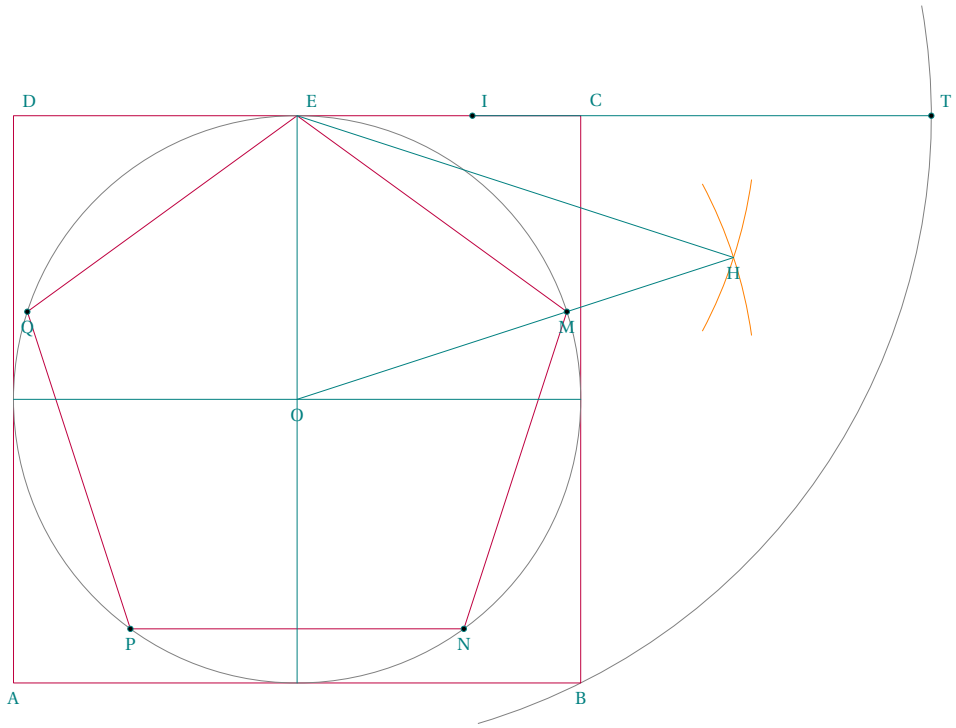



```
\begin{tikzpicture}[scale=.75]
   \tkzDefPoint(0,0){O}
   \tkzDefPoint(5,0){A}
   \tkzDefPoint(0,5){B}
   \tkzDefPoint(-5,0){C}
   \tkzDefPoint(0,-5){D}
   \tkzDefMidPoint(A,O)              \tkzGetPoint{I}
   \tkzInterLC(I,B)(I,A)             \tkzGetPoints{F}{E}
   \tkzInterCC(O,C)(B,E)             \tkzGetPoints{D3}{D2}
   \tkzInterCC(O,C)(B,F)             \tkzGetPoints{D4}{D1}
   \tkzDrawArc[angles](B,E)(180,360)
   \tkzDrawArc[angles](B,F)(220,340)
   \tkzDrawLine[add=.5 and .5](B,I)
   \tkzDrawCircle(O,A)
   \tkzDefCircle[diameter](O,A)      \tkzGetPoint{x}
   \tkzDrawCircle(x,A)
   \tkzDrawSegments(B,D C,A)
   \tkzDrawPolygon[new](D,D1,D2,D3,D4)
   \tkzDrawPoints(A,...,D,O)
   \tkzDrawPoints[new](E,F,I,D1,D2,D4,D3)
   \tkzLabelPoints[below left](A,...,D,O)
   \tkzLabelPoints[new,below right](I,E,F,D1,D2,D4,D3)
\end{tikzpicture}
```

51.16. Pentagon in a square

Pentagon in a square

To inscribe a pentagon in a given square.



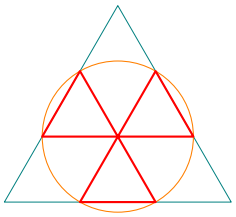

```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoints{0/0/O,-5/-5/A,5/-5/B}
  \tkzDefSquare(A,B)    \tkzGetPoints{C}{D}
  \tkzDefMidPoint(A,B) \tkzGetPoint{F}
  \tkzDefMidPoint(C,D) \tkzGetPoint{E}
  \tkzDefMidPoint(B,C) \tkzGetPoint{G}
  \tkzDefMidPoint(A,D) \tkzGetPoint{K}
  \tkzInterLC[near](C,D)(E,B)                \tkzGetFirstPoint{T}
  \tkzDefMidPoint(D,T)                       \tkzGetPoint{I}
  \tkzInterCC[with nodes](O,D,I)(E,D,I)      \tkzGetSecondPoint{H}
  \tkzInterLC(O,H)(O,E)                      \tkzGetSecondPoint{M}
  \tkzInterCC(O,E)(E,M)                      \tkzGetFirstPoint{Q}
  \tkzInterCC[with nodes](O,O,E)(Q,E,M)      \tkzGetFirstPoint{P}
  \tkzInterCC[with nodes](O,O,E)(P,E,M)      \tkzGetFirstPoint{N}
  \tkzCompass(O,H E,H)
  \tkzDrawArc(E,B)(T)
  \tkzDrawPolygons[purple](A,B,C,D M,E,Q,P,N)
  \tkzDrawCircle(O,E)
  \tkzDrawSegments(T,I O,H E,H E,F G,K)
  \tkzDrawPoints(T,M,Q,P,N,I)
  \tkzLabelPoints(A,B,O,N,P,Q,M,H)
  \tkzLabelPoints[above right](C,D,E,I,T)
\end{tikzpicture}
```

51.17. Inscribed hexagon

Hexagon Inscribed

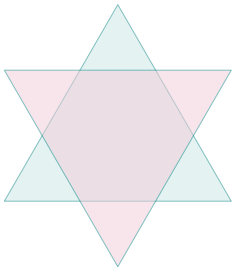
To inscribe a regular hexagon in a given equilateral triangle perfectly inside it (no boarders).

51.17.1. Inscribed hexagon: version 1



```
\begin{tikzpicture}[scale=.5]
 \pgfmathsetmacro{\c}{6}
 \tkzDefPoints{0/0/A,\c/0/B}
 \tkzDefTriangle[equilateral](A,B)\tkzGetPoint{C}
 \tkzDefTriangleCenter[centroid](A,B,C)
 \tkzGetPoint{I}
 \tkzDefPointBy[homothety=center A ratio 1./3](B)
 \tkzGetPoint{c1}
 \tkzInterLC(B,C)(I,c1) \tkzGetPoints{a1}{a2}
 \tkzInterLC(A,C)(I,c1) \tkzGetPoints{b1}{b2}
 \tkzInterLC(A,B)(I,c1) \tkzGetPoints{c1}{c2}
 \tkzDrawPolygon(A,B,C)
 \tkzDrawCircle[thin,orange](I,c1)
 \tkzDrawPolygon[red,thick](a2,a1,b2,b1,c2,c1)
\end{tikzpicture}
```

51.17.2. Inscribed hexagon: version 2

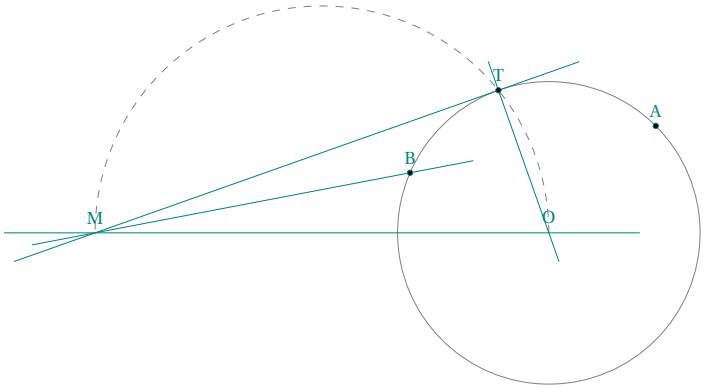


```
\begin{tikzpicture}[scale=.5]
 \pgfmathsetmacro{\c}{6}
 \tkzDefPoints{0/0/A,\c/0/B}
 \tkzDefTriangle[equilateral](A,B)\tkzGetPoint{C}
 \tkzDefTriangleCenter[centroid](A,B,C)
 \tkzGetPoint{I}
 \tkzDefPointsBy[rotation= center I
                 angle 60](A,B,C){a,b,c}
 \tkzDrawPolygon[fill=teal!20,opacity=.5](A,B,C)
 \tkzDrawPolygon[fill=purple!20,opacity=.5](a,b,c)
\end{tikzpicture}
```

51.18. Power of a point with respect to a circle

Power of a point with respect to a circle

$\overline{MA} \times \overline{MB} = MT^2 = MO^2 - OT^2$

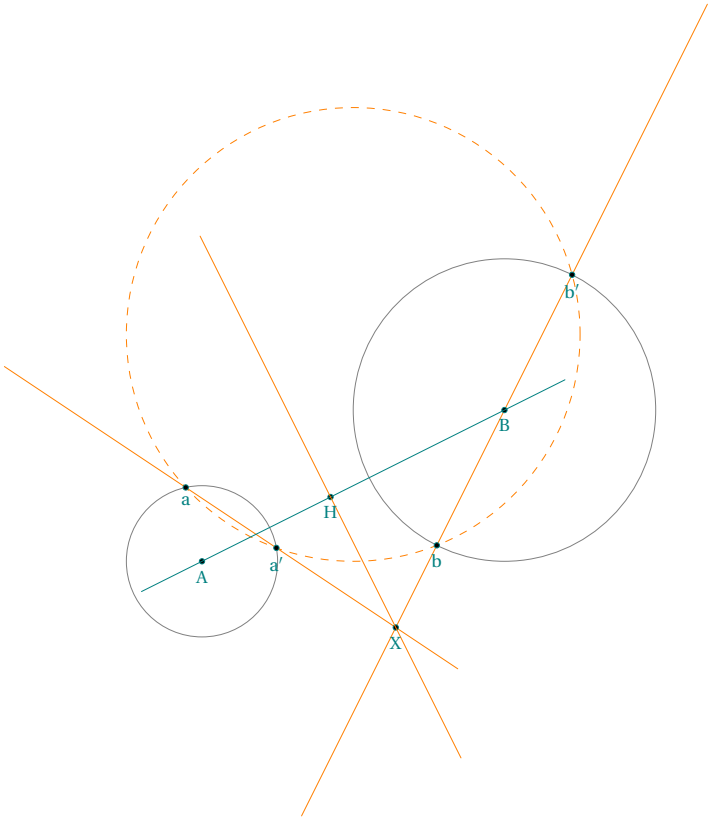



```
\begin{tikzpicture}
 \pgfmathsetmacro{\r}{2}%
 \pgfmathsetmacro{\xO}{6}%
 \pgfmathsetmacro{\xE}{\xO-\r}%
 \tkzDefPoints{0/0/M,\xO/0/O,\xE/0/E}
 \tkzDefCircle[diameter](M,O)
 \tkzGetPoint{I}
 \tkzInterCC(I,O)(O,E) \tkzGetPoints{T}{T'}
 \tkzDefShiftPoint[O](45:2){B}
 \tkzInterLC(M,B)(O,E) \tkzGetPoints{A}{B}
 \tkzDrawCircle(O,E)
 \tkzDrawSemiCircle[dashed](I,O)
 \tkzDrawLine(M,O)
 \tkzDrawLines(M,T O,T M,B)
 \tkzDrawPoints(A,B,T)
 \tkzLabelPoints[above](A,B,O,M,T)
\end{tikzpicture}
```

51.19. Radical axis of two non-concentric circles

Radical axis of two non-concentric circles

From Wikipedia : *In geometry, the radical axis of two non-concentric circles is the set of points whose power with respect to the circles are equal. For this reason the radical axis is also called the power line or power bisector of the two circles. The notation radical axis was used by the French mathematician M. Chasles as axe radical.*

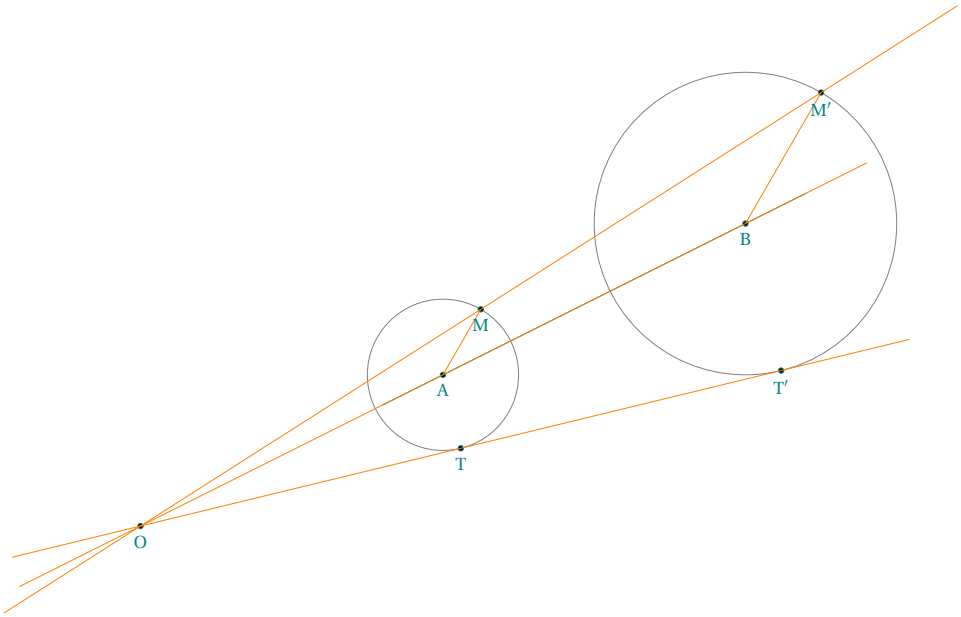



```
\begin{tikzpicture}
\tkzDefPoints{0/0/A,4/2/B,2/3/K}
\tkzDefCircle[R](A,1)\tkzGetPoint{a}
\tkzDefCircle[R](B,2)\tkzGetPoint{b}
\tkzDefCircle[R](K,3)\tkzGetPoint{k}
\tkzDrawCircles(A,a B,b)
\tkzDrawCircle[dashed,new](K,k)
\tkzInterCC(A,a)(K,k) \tkzGetPoints{a}{a'}
\tkzInterCC(B,b)(K,k) \tkzGetPoints{b}{b'}
\tkzDrawLines[new,add=2 and 2](a,a')
\tkzDrawLines[new,add=1 and 1](b,b')
\tkzInterLL(a,a')(b,b') \tkzGetPoint{X}
\tkzDefPointBy[projection= onto A--B](X) \tkzGetPoint{H}
\tkzDrawPoints(A,B,H,X,a,b,a',b')
\tkzDrawLine(A,B)
\tkzDrawLine[add= 1 and 2,new](X,H)
\tkzLabelPoints(A,B,H,X,a,b,a',b')
\end{tikzpicture}
```

51.20. External homothetic center

External homothetic center

From Wikipedia : *Given two nonconcentric circles, draw radii parallel and in the same direction. Then the line joining the extremities of the radii passes through a fixed point on the line of centers which divides that line externally in the ratio of radii. This point is called the external homothetic center, or external center of similitude (Johnson 1929, pp. 19-20 and 41).*

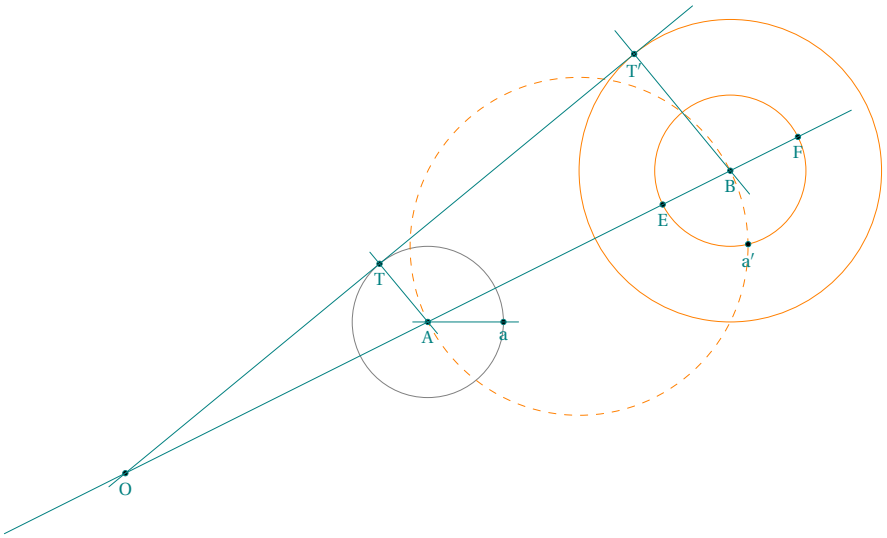



```
\begin{tikzpicture}
\tkzDefPoints{0/0/A,4/2/B,2/3/K}
\tkzDefCircle[R](A,1)\tkzGetPoint{a}
\tkzDefCircle[R](B,2)\tkzGetPoint{b}
\tkzDrawCircles(A,a B,b)
\tkzDrawLine(A,B)
\tkzDefShiftPoint[A](60:1){M}
\tkzDefShiftPoint[B](60:2){M'}
\tkzInterLL(A,B)(M,M') \tkzGetPoint{O}
\tkzDefLine[tangent from = O](B,M') \tkzGetPoints{X}{T'}
\tkzDefLine[tangent from = O](A,M) \tkzGetPoints{X}{T}
\tkzDrawPoints(A,B,O,T,T',M,M')
\tkzDrawLines[new](O,B O,T' O,M')
\tkzDrawSegments[new](A,M B,M')
\tkzLabelPoints(A,B,O,T,T',M,M')
\end{tikzpicture}
```

51.21. Tangent lines to two circles

Tangent lines to two circles

For two circles, there are generally four distinct lines that are tangent to both if the two circles are outside each other. For two of these, the external tangent lines, the circles fall on the same side of the line; the external tangent lines intersect in the external homothetic center

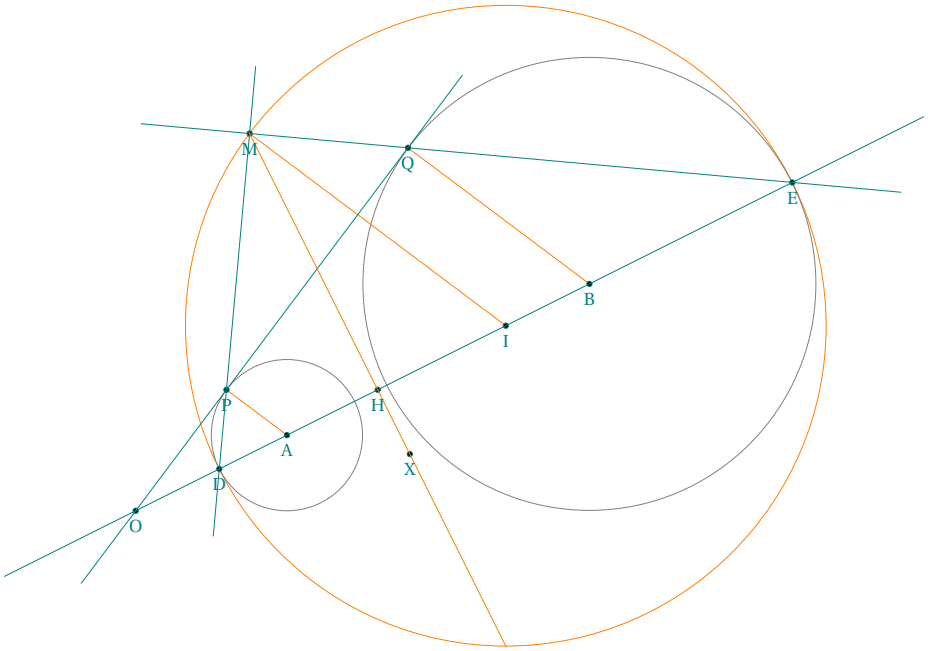



```
\begin{tikzpicture}
 \pgfmathsetmacro{\r}{1}%
 \pgfmathsetmacro{\R}{2}%
 \pgfmathsetmacro{\rt}{\R-\r}%
 \tkzDefPoints{0/0/A,4/2/B,2/3/K}
 \tkzDefMidPoint(A,B) \tkzGetPoint{I}
 \tkzInterLC[R](A,B)(B,\rt) \tkzGetPoints{E}{F}
 \tkzInterCC(I,B)(B,F) \tkzGetPoints{a}{a'}
 \tkzInterLC[R](B,a)(B,\R) \tkzGetPoints{X'}{T'}
 \tkzDefLine[tangent at=T'](B) \tkzGetPoint{h}
 \tkzInterLL(T',h)(A,B) \tkzGetPoint{O}
 \tkzInterLC[R](O,T')(A,\r) \tkzGetPoints{T}{T}
 \tkzDefCircle[R](A,\r)  \tkzGetPoint{a}
 \tkzDefCircle[R](B,\R)  \tkzGetPoint{b}
 \tkzDefCircle[R](B,\rt)  \tkzGetPoint{c}
 \tkzDrawCircles(A,a)
 \tkzDrawCircles[orange](B,b B,c)
 \tkzDrawCircle[orange,dashed](I,B)
 \tkzDrawPoints(O,A,B,a,a',E,F,T',T)
 \tkzDrawLines(O,B A,a B,T' A,T)
 \tkzDrawLines[add= 1 and 8](T',h)
 \tkzLabelPoints(O,A,B,a,a',E,F,T,T')
\end{tikzpicture}
```

51.22. Tangent lines to two circles with radical axis

Tangent lines to two circles with radical axis

As soon as two circles are not concentric, we can construct their radical axis, the set of points of equal power with respect to the two circles. We know that the radical axis is a line orthogonal to the line of the centers. Note that if we specify P *and* Q *as the points of contact of one of the common exterior tangents with the two circles and* D *and* E *as the points of the circles outside [AB], then (DP) and (EQ) intersect on the radical axis of the two circles. We will show that this property is always true and that it allows us to construct common tangents, even when the circles have the same radius.*

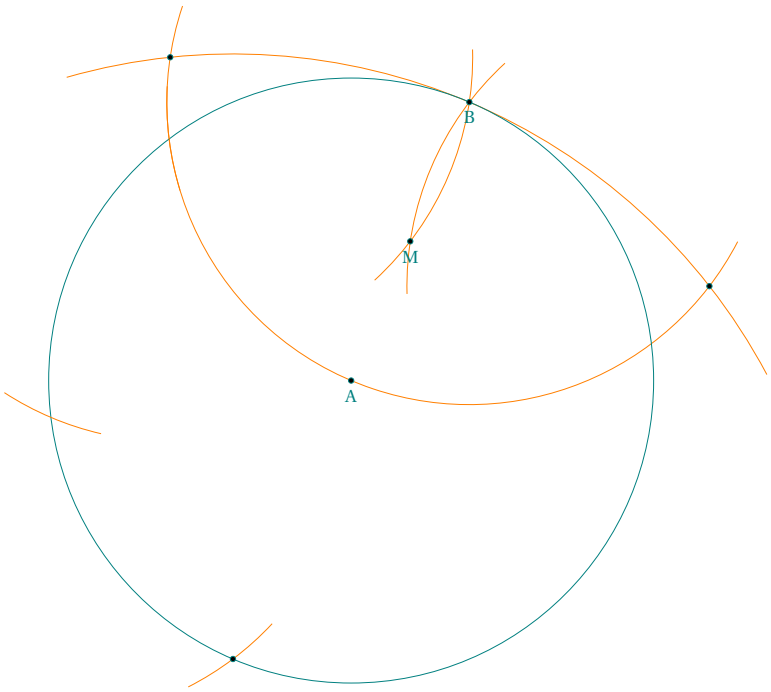


```
\begin{tikzpicture}
\tkzDefPoints{0/0/A,4/2/B,2/3/K}
\tkzDefCircle[R](A,1) \tkzGetPoint{a}
\tkzDefCircle[R](B,3) \tkzGetPoint{b}
\tkzInterCC[R](A,1)(K,3) \tkzGetPoints{a}{a'}
\tkzInterCC[R](B,3)(K,3) \tkzGetPoints{b}{b'}
\tkzInterLL(a,a')(b,b')  \tkzGetPoint{X}
\tkzDefPointBy[projection= onto A--B](X) \tkzGetPoint{H}
\tkzGetPoint{C}
\tkzInterLC[R](A,B)(B,3) \tkzGetPoints{b1}{E}
\tkzInterLC[R](A,B)(A,1) \tkzGetPoints{D}{a2}
\tkzDefMidPoint(D,E) \tkzGetPoint{I}
\tkzDrawCircle[orange](I,D)
\tkzInterLC(X,H)(I,D) \tkzGetPoints{M}{M'}
\tkzInterLC(M,D)(A,D) \tkzGetPoints{P}{P'}
\tkzInterLC(M,E)(B,E) \tkzGetPoints{Q'}{Q}
\tkzInterLL(P,Q)(A,B) \tkzGetPoint{O}
\tkzDrawCircles(A,a B,b)
\tkzDrawSegments[orange](A,P I,M B,Q)
\tkzDrawPoints(A,B,D,E,M,I,O,P,Q,X,H)
\tkzDrawLines(O,E M,D M,E O,Q)
\tkzDrawLine[add= 3 and 4,orange](X,H)
\tkzLabelPoints(A,B,D,E,M,I,O,P,Q,X,H)
\end{tikzpicture}
```

51.23. Midpoint of a segment with a compass

Tangent lines to two circles with radical axis

This example involves determining the midpoint of a segment, using only a compass.



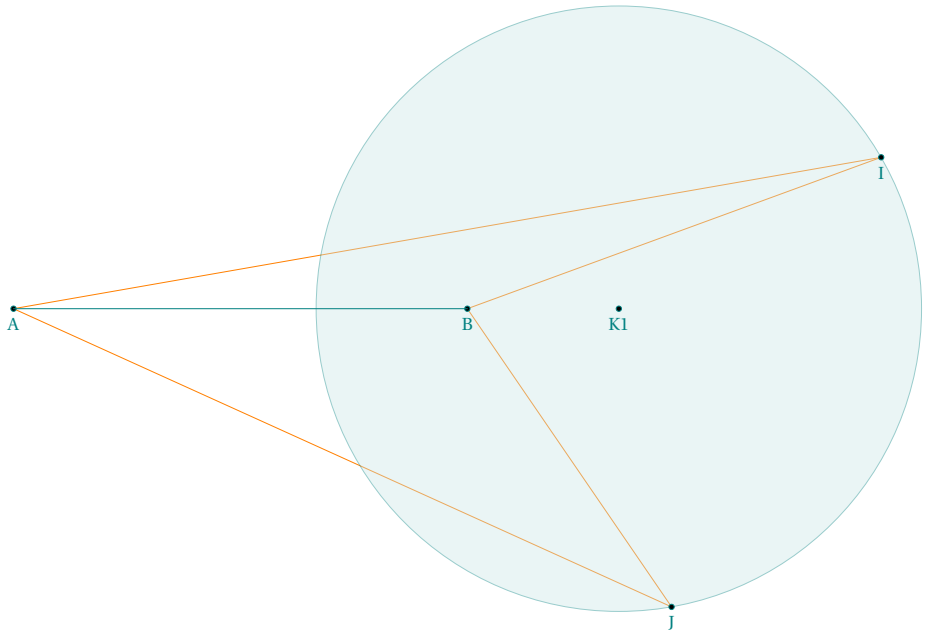

```
\begin{tikzpicture}
\tkzDefPoint(0,0){A}
\tkzDefRandPointOn[circle= center A radius 4]    \tkzGetPoint{B}
\tkzDefPointBy[rotation= center A angle 180](B)  \tkzGetPoint{C}
\tkzInterCC(A,B)(B,A)                           \tkzGetPoints{I}{I'}
\tkzInterCC(A,I)(I,A)                           \tkzGetPoints{J}{B}
\tkzInterCC(B,A)(C,B)                           \tkzGetPoints{D}{E}
\tkzInterCC(D,B)(E,B)                           \tkzGetPoints{M}{M'}
\tkzSetUpArc[color=orange,style=solid,delta=10]
\tkzDrawArc(C,D)(E)
\tkzDrawArc(B,E)(D)
\tkzDrawCircle[color=teal,line width=.2pt](A,B)
\tkzDrawArc(D,B)(M)
\tkzDrawArc(E,M)(B)
\tkzCompasss[color=orange,style=solid](B,I I,J J,C)
\tkzDrawPoints(A,B,C,D,E,M)
\tkzLabelPoints(A,B,M)
\end{tikzpicture}
```

51.24. Definition of a circle, _Apollonius_

Definition of a circle _Apollonius_

From Wikipedia : *Apollonius showed that a circle can be defined as the set of points in a plane that have a specified ratio of distances to two fixed points, known as foci. This Apollonian circle is the basis of the Apollonius pursuit problem. ... The solutions to this problem are sometimes called the circles of Apollonius.*

Explanation
A circle is the set of points of a plane that are equidistant from a given point O. The distance r from the center is called the radius, and the point O is called the center. It is the simplest definition but it is not the only one. Apollonius of Perga gives another definition: The set of all points whose distances from two fixed points are in a constant ratio is a circle.
With `tkz-euclide` it is easy to illustrate the last definition.

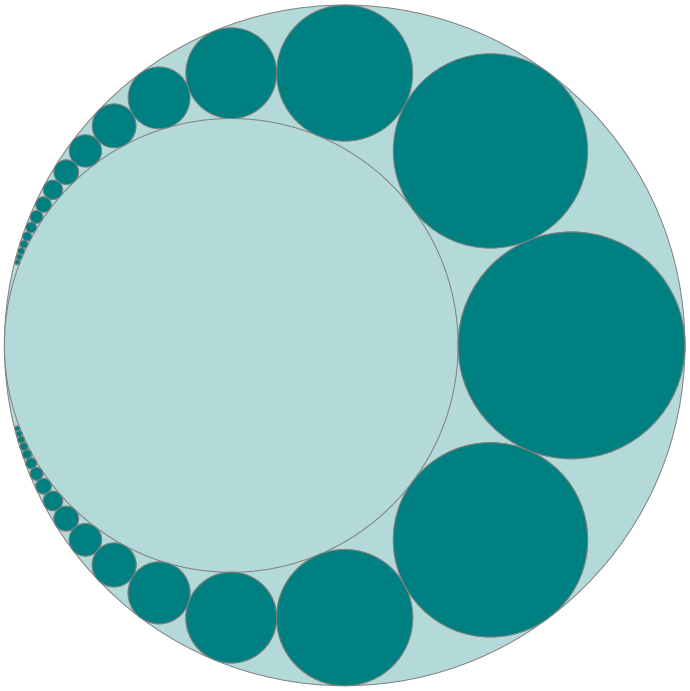



```
\begin{tikzpicture}[scale=1.5]
    % Firstly we defined two fixed point.
    % The figure depends of these points and the ratio K
\tkzDefPoint(0,0){A}
\tkzDefPoint(4,0){B}
    % tkz-euclide.sty knows about the apollonius's circle
    % with K=2 we search some points like  I such as IA=2 x IB
\tkzDefCircle[apollonius,K=2](A,B) \tkzGetPoints{K1}{k}
\tkzDefPointOnCircle[through=  center K1 angle 30 point k]
\tkzGetPoint{I}
\tkzDefPointOnCircle[through= center K1 angle 280  point k]
\tkzGetPoint{J}
\tkzDrawSegments[new](A,I I,B A,J J,B)
\tkzDrawCircle[color = teal,fill=teal!20,opacity=.4](K1,k)
\tkzDrawPoints(A,B,K1,I,J)
\tkzDrawSegment(A,B)
\tkzLabelPoints[below,font=\scriptsize](A,B,K1,I,J)
\end{tikzpicture}
```

51.25. Application of Inversion: Pappus chain

Pappus chain

From Wikipedia *In geometry, the Pappus chain is a ring of circles between two tangent circles investigated by Pappus of Alexandria in the 3rd century AD.*

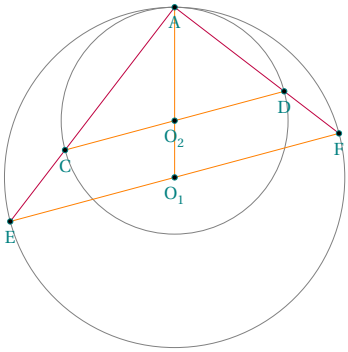


```
\begin{tikzpicture}[ultra thin]
  \pgfmathsetmacro{\xB}{6}%
  \pgfmathsetmacro{\xC}{9}%
  \pgfmathsetmacro{\xD}{(\xC*\xC)/\xB}%
  \pgfmathsetmacro{\xJ}{(\xC+\xD)/2}%
  \pgfmathsetmacro{\r}{\xD-\xJ}%
  \pgfmathsetmacro{\nc}{16}%
  \tkzDefPoints{0/0/A,\xB/0/B,\xC/0/C,\xD/0/D}
  \tkzDefCircle[diameter](A,C) \tkzGetPoint{x}
  \tkzDrawCircle[fill=teal!30](x,C)
  \tkzDefCircle[diameter](A,B) \tkzGetPoint{y}
  \tkzDrawCircle[fill=teal!30](y,B)
  \foreach \i in {-\nc,...,0,...,\nc}
  {\tkzDefPoint(\xJ,2*\r*\i){J}
   \tkzDefPoint(\xJ,2*\r*\i-\r){H}
   \tkzDefCircleBy[inversion = center A through C](J,H)
   \tkzDrawCircle[fill=teal](tkzFirstPointResult,tkzSecondPointResult)}
\end{tikzpicture}
```

51.26. Archimedes, Book of Lemmas, Proposition 1

Book of lemmas proposition 1 Archimedes

If two circles touch at A, *and if* [CD], [EF] *be parallel diameters in them,* A, C *and* E *are aligned.*




```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoints{0/0/O_1,0/1/O_2,0/3/A}
  \tkzDefPoint(15:3){F}
   \tkzDefPointBy[symmetry=center O_1](F) \tkzGetPoint{E}
   \tkzDefLine[parallel=through O_2](E,F) \tkzGetPoint{x}
   \tkzInterLC(x,O_2)(O_2,A) \tkzGetPoints{D}{C}
   \tkzDrawCircles(O_1,A O_2,A)
   \tkzDrawSegments[orange](O_1,A E,F C,D)
   \tkzDrawSegments[purple](A,E A,F)
   \tkzDrawPoints(A,O_1,O_2,E,F,x,C,D)
   \tkzLabelPoints(A,O_1,O_2,E,F,x,C,D)
\end{tikzpicture}
```

(CD) ∥ (EF) and (AO_1) intersects these two lines, so $\widehat{AO_2C} = \widehat{AO_1E}$.
Since the triangles AO_2C and AO_1E are isosceles, their base angles are equal: $\widehat{ACO_2} = \widehat{AEO_1} = \widehat{CAO_2} = \widehat{EAO_1}$. Thus A, C and E are aligned.

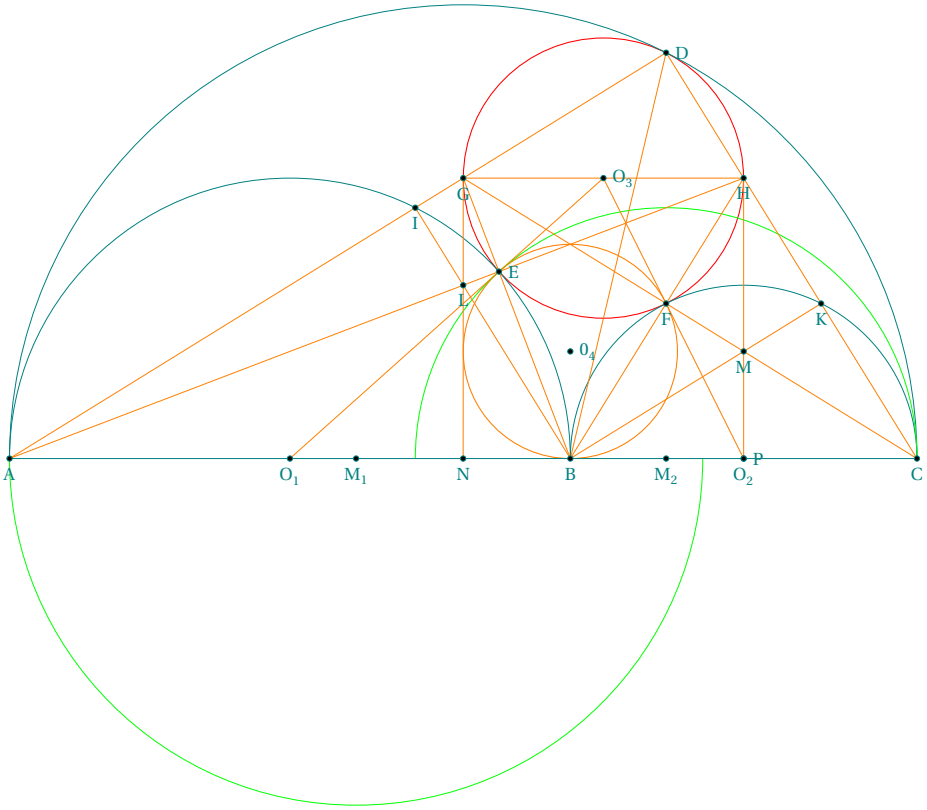
51.27. Archimedes, Book of Lemmas, Proposition 6

Book of lemmas proposition 6 Archimedes

Let AC, *the diameter of a semicircle, be divided at* B *so that* AC/AB = ϕ *or in any ratio. Describe semicircles within the first semicircle and on* AB, BC *as diameters, and suppose a circle drawn touching the all three semicircles. If* GH *be the diameter of this circle, to find relation between* GH *and* AC.

```
\begin{tikzpicture}
\tkzDefPoints{0/0/A,12/0/C}
\tkzDefGoldenRatio(A,C)                    \tkzGetPoint{B}
\tkzDefMidPoint(A,C)                       \tkzGetPoint{O}
\tkzDefMidPoint(A,B)                       \tkzGetPoint{O_1}
\tkzDefMidPoint(B,C)                       \tkzGetPoint{O_2}
\tkzDefExtSimilitudeCenter(O_1,A)(O_2,B)   \tkzGetPoint{M_0}
\tkzDefIntSimilitudeCenter(O,A)(O_1,A)     \tkzGetPoint{M_1}
\tkzDefIntSimilitudeCenter(O,C)(O_2,C)     \tkzGetPoint{M_2}
\tkzInterCC(O_1,A)(M_2,C)                  \tkzGetFirstPoint{E}
\tkzInterCC(O_2,C)(M_1,A)                  \tkzGetSecondPoint{F}
\tkzInterCC(O,A)(M_0,B)                    \tkzGetFirstPoint{D}
\tkzInterLL(O_1,E)(O_2,F)                  \tkzGetPoint{O_3}
\tkzDefCircle[circum](E,F,B)               \tkzGetPoint{0_4}
\tkzInterLC(A,D)(O_1,A)                    \tkzGetFirstPoint{I}
\tkzInterLC(C,D)(O_2,B)                    \tkzGetSecondPoint{K}
\tkzInterLC[common=D](A,D)(O_3,D)          \tkzGetFirstPoint{G}
\tkzInterLC[common=D](C,D)(O_3,D)          \tkzGetFirstPoint{H}
\tkzInterLL(C,G)(B,K)                      \tkzGetPoint{M}
\tkzInterLL(A,H)(B,I)                      \tkzGetPoint{L}
\tkzInterLL(L,G)(A,C)                      \tkzGetPoint{N}
\tkzInterLL(M,H)(A,C)                      \tkzGetPoint{P}
\tkzDrawCircles[red,thin](O_3,F)
\tkzDrawCircles[new,thin](0_4,B)
```

```
\tkzDrawSemiCircles[teal](O,C O_1,B O_2,C)
\tkzDrawSemiCircles[green](M_2,C)
\tkzDrawSemiCircles[green,swap](M_1,A)
\tkzDrawSegment(A,C)
\tkzDrawSegments[new](O_1,O_3 O_2,O_3)
\tkzDrawSegments[new,very thin](B,H C,G A,H G,N H,P)
\tkzDrawSegments[new,very thin](B,D A,D C,D G,H I,B K,B B,G)
\tkzDrawPoints(A,B,C,M_1,M_2,E,O_3,F,D,O_4,O_1,O_2,I,K,G,H,L,P,N,M)
\tkzLabelPoints[font=\scriptsize](A,B,C,M_1,M_2,F,O_1,O_2,I,K,G,H,L,M,N)
\tkzLabelPoints[font=\scriptsize,right](E,O_3,D,O_4,P)
\end{tikzpicture}
```



Let GH be the diameter of the circle which is parallel to AC, and let the circle touch the semicircles on AC, AB, BC in D, E, F respectively.
Then, by Prop. 1, A, G and D are aligned, as well as D, H and C.
For the same reason, A, E and H are aligned, C, F and G are aligned, and so are B, E, G and B, F, H.
Let (AD) meet the semicircle on [AC] at I, and let (BD) meet the semicircle on [BC] in K. Join CI and CK, meeting AE and BF in L and M, and let GL and HM, extended, meet AB in N and P respectively.
Now, in the triangle AGB, the perpendiculars from A, C on the opposite sides meet in L. Therefore by the properties of triangles, (GN) is perpendicular to (AC). Similarly (HP) is perpendicular to (BC).
Again, since the angles at I, K, D are right, (CK) is parallel to (AD), and (CI) to (BD).
Therefore

$$\frac{AB}{BC}=\frac{AL}{LH}=\frac{AN}{NP}\quad\text{and}\quad\frac{BC}{AB}=\frac{CM}{MG}=\frac{PC}{NP}$$

hence

$$\frac{AN}{NP}=\frac{NP}{PC}\quad\text{so}\quad NP^2=AN\times PC$$

Now suppose that B divides [AC] in the golden ratio, that is:

$$\phi = \frac{AB}{BC} = \frac{AC}{AB} \quad \text{then} \quad AN = \phi NP \text{and} \quad NP = \phi PC$$

We have

$$AC = AN + NP + PC \quad \text{either} \quad AB + BC == AN + NP + PC \quad \text{or} \quad (\phi + 1)BC = AN + NP + PC$$

we get

$$(\phi + 1)BC = \phi NP + NP + PC = (\phi + 1)NP + PC = \phi(\phi + 1)PC + PC = \phi^2 + \phi + 1)PC$$

as

$$\phi^2 = \phi + 1 \quad \text{then} \quad (\phi + 1)BC = 2(\phi + 1)PC \quad \text{i.e.} \quad BC = 2PC$$

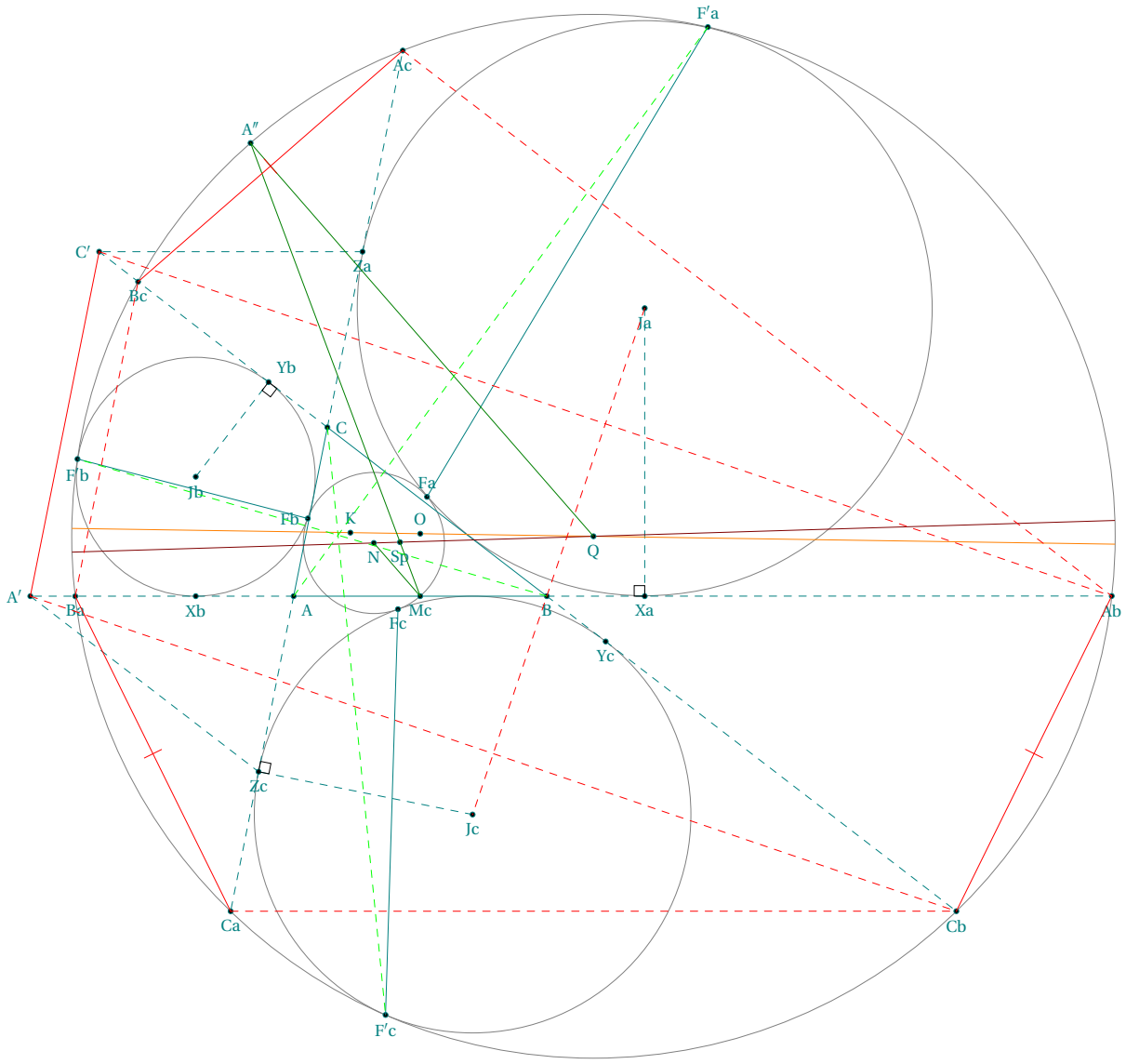
That is, p is the midpoint of the segment BC.
Part of the proof from https://www.cut-the-knot.org

51.28. "The" circle of Apollonius

The Apollonius circle of a triangle _Apollonius_

The circle which touches all three excircles of a triangle and encompasses them is often known as "the" Apollonius circle (Kimberling 1998, p. 102)

Explanation
The purpose of the first examples was to show how simply these propositions can be recreated. With TikZ you need calculations and trigonometry, while with `tkz-euclide` you only need to build simple objects.
But don't forget that TikZ is behind `tkz-euclide`, and far above it: I am only creating an interface between TikZ and the users of my package.
The last example is very complex; it shows everything that can be done with `tkz-euclide`.




```
\begin{tikzpicture}[scale=.6]
\tkzDefPoints{0/0/A,6/0/B,0.8/4/C}
\tkzDefTriangleCenter[euler](A,B,C)         \tkzGetPoint{N}
\tkzDefTriangleCenter[circum](A,B,C)        \tkzGetPoint{O}
\tkzDefTriangleCenter[lemoine](A,B,C)       \tkzGetPoint{K}
\tkzDefTriangleCenter[ortho](A,B,C)         \tkzGetPoint{H}
\tkzDefSpcTriangle[excentral,name=J](A,B,C){a,b,c}
\tkzDefSpcTriangle[centroid,name=M](A,B,C){a,b,c}
\tkzDefCircle[in](Ma,Mb,Mc)                 \tkzGetPoint{Sp}  % Sp Spieker center
\tkzDefProjExcenter[name=J](A,B,C)(a,b,c){Y,Z,X}
\tkzDefLine[parallel=through Za](A,B)       \tkzGetPoint{Xc}
\tkzInterLL(Za,Xc)(C,B)                     \tkzGetPoint{C'}
\tkzDefLine[parallel=through Zc](B,C)       \tkzGetPoint{Ya}
\tkzInterLL(Zc,Ya)(A,B)                     \tkzGetPoint{A'}
\tkzDefPointBy[reflection= over Ja--Jc](C')\tkzGetPoint{Ab}
\tkzDefPointBy[reflection= over Ja--Jc](A')\tkzGetPoint{Cb}
\tkzInterLL(K,O)(N,Sp)                      \tkzGetPoint{Q}
```

```
\tkzInterLC(A,B)(Q,Cb)                    \tkzGetFirstPoint{Ba}
\tkzInterLC(A,C)(Q,Cb)                    \tkzGetPoints{Ac}{Ca}
\tkzInterLC(B,C')(Q,Cb)                   \tkzGetFirstPoint{Bc}
\tkzInterLC[next to=Ja](Ja,Q)(Q,Cb)       \tkzGetFirstPoint{F'a}
\tkzInterLC[next to=Jc](Jc,Q)(Q,Cb)       \tkzGetFirstPoint{F'c}
\tkzInterLC[next to=Jb](Jb,Q)(Q,Cb)       \tkzGetFirstPoint{F'b}
\tkzInterLC[common=F'a](Sp,F'a)(Ja,F'a)   \tkzGetFirstPoint{Fa}
\tkzInterLC[common=F'b](Sp,F'b)(Jb,F'b)   \tkzGetFirstPoint{Fb}
\tkzInterLC[common=F'c](Sp,F'c)(Jc,F'c)   \tkzGetFirstPoint{Fc}
\tkzInterLC(Mc,Sp)(Q,Cb)                  \tkzGetFirstPoint{A''}
\tkzDefCircle[euler](A,B,C)               \tkzGetPoints{E}{e}
\tkzDefCircle[ex](C,A,B)                  \tkzGetPoints{Ea}{a}
\tkzDefCircle[ex](A,B,C)                  \tkzGetPoints{Eb}{b}
\tkzDefCircle[ex](B,C,A)                  \tkzGetPoints{Ec}{c}
% Calculations are done, now you can draw, mark and label
\tkzDrawCircles(Q,Cb E,e)%
\tkzDrawCircles(Eb,b Ea,a Ec,c)
\tkzDrawPolygon(A,B,C)
\tkzDrawSegments[dashed](A,A' C,C' A',Zc Za,C' B,Cb B,Ab A,Ca)
\tkzDrawSegments[dashed](C,Ac Ja,Xa Jb,Yb Jc,Zc)
\begin{scope}
   \tkzClipCircle(Q,Cb) % We limit the drawing of the lines
   \tkzDrawLine[add=5 and 12,orange](K,O)
   \tkzDrawLine[add=12 and 28,red!50!black](N,Sp)
\end{scope}
\tkzDrawPoints(A,B,C,K,Ja,Jb,Jc,Q,N,O,Sp,Mc,Xa,Xb,Yb,Yc,Za,Zc)
\tkzDrawPoints(A',C',A'',Ab,Cb,Bc,Ca,Ac,Ba,Fa,Fb,Fc,F'a,F'b,F'c)
\tkzLabelPoints(Ja,Jb,Jc,Q,Xa,Xb,Za,Zc,Ab,Cb,Bc,Ca,Ac,Ba,F'b)
\tkzLabelPoints[above](O,K,F'a,Fa,A'')
\tkzLabelPoints[below](B,F'c,Yc,N,Sp,Fc,Mc)
\tkzLabelPoints[left](A',C',Fb)
\tkzLabelPoints[right](C)
\tkzLabelPoints[below right](A)
\tkzLabelPoints[above right](Yb)
\tkzDrawSegments(Fc,F'c Fb,F'b Fa,F'a)
\tkzDrawSegments[color=green!50!black](Mc,N Mc,A'' A'',Q)
\tkzDrawSegments[color=red,dashed](Ac,Ab Ca,Cb Ba,Bc Ja,Jc A',Cb C',Ab)
\tkzDrawSegments[color=red](Cb,Ab Bc,Ac Ba,Ca A',C')
\tkzMarkSegments[color=red,mark=|](Cb,Ab Bc,Ac Ba,Ca)
\tkzMarkRightAngles(Jc,Zc,A Ja,Xa,B Jb,Yb,C)
\tkzDrawSegments[green,dashed](A,F'a B,F'b C,F'c)
\end{tikzpicture}
```

Part XI.

FAQ

52. FAQ

52.1. Most common errors

These are the mistakes I make most often myself, since the syntax has changed several times. This section will be expanded.

- Forgetting the "s" of the macros that draw several objects, such as `\tkzDrawSegments` or `\tkzDrawCircles`: `\tkzDrawPoint(A,B)` is wrong, you need `\tkzDrawPoints(A,B)`.

- Since version 4, the unit is always the cm, so you must not write a unit: `\tkzDefCircle[R](O,3cm)` becomes `\tkzDefCircle[R](O,3)`. The usual `TikZ` options keep their units, for example `below right = 12pt`; on the other hand, you write `size=1.2` to set the radius of the arc in `\tkzMarkAngle`.

- Parentheses instead of braces. A point that is created has its name between braces, while a point that is used, as an option or as an argument, has its name between parentheses: write `\tkzGetPoint{A}`, not `\tkzGetPoint(A)`.

- `\tkzGetPoint{A}` instead of `\tkzGetFirstPoint{A}`. When a macro gives two points, either retrieve both with `\tkzGetPoints{A}{B}`, or only one of them with `\tkzGetFirstPoint{A}` or `\tkzGetSecondPoint{A}`. These points can also be used with the references `tkzFirstPointResult` and `tkzSecondPointResult`. Some macros also give a third point, `tkzPointResult`.

- Retrieve a result immediately: `\tkzGetPoint`, `\tkzGetPoints` or `\tkzGetLength` must follow the macro that computes the result, because the next calculation replaces it.

- The order of the intersection points of a line and a circle, or of two circles, has sometimes changed between versions; this leads either to a wrong figure or to an error. The options `common` and `near` let you choose the point you want.

- Mixing options and arguments: all the macros that use a circle need to know its radius. If the radius is given by a measure, the option includes `R`.

- Angles are given in degrees, more rarely in radians.

- A list of points must not be cut by a `%` at the end of a line: `\tkzDrawSegments(A,B%` followed by `C,D)` gives the point `BC,D`, because `%` also removes the space between the two points. Simply go to the next line without `%`.

- The syntax of version 3 for special circles, for example `\tkzDrawCircle[euler](A,B,C)`, no longer exists: define the circle with `\tkzDefCircle[euler](A,B,C)`, retrieve its center and a point with `\tkzGetPoints{N}{P}`, then draw it with `\tkzDrawCircle(N,P)`. Since version 5.14, the old syntax gives an error message.

- If an error occurs in a calculation when passing arguments, it is better to do the calculation before calling the macro.

- Do not mix the syntaxes of `pgfmath` and `xfp`. I have often chosen `xfp`; if you prefer `pgfmath`, do your calculations before passing the arguments.

- The error "dimension too large" occurs in some cases, for example with `\tkzDrawArc` or `\tkzMarkAngle` and a very small angle: it comes from the library `decorations` when a mark is placed on the arc, even

when the mark is absent. One way to avoid it is the option **veclen** (see section 31.6): in the scope where it is used, the function veclen is computed with xfp. Do not use intersection macros in this scope.

– Since version 5.14, **\tkzSetUpCompass**[delta=...] no longer extends the arcs drawn with **\tkzDrawArc**: use **\tkzSetUpArc**[delta=...] for them.

Part XII.

News, compatibility and options

53. News and compatibility

53.1. With version 5.15

– \tkzInterLC with a line passing through the center of the circle: the order of the two points no longer depends on rounding. **tkzSecondPoint** is the point located in the direction of the line (from its first point to its second point); the results are the same with and without the option **lua**. See section 19.2.3.

53.2. With version 5.14

This version only corrects bugs; the syntax does not change and no macro has been added.

– \tkzDefCircle with the options **orthogonal from** or **orthogonal through** no longer disturbs the following calls of \tkzDefCircle.

– The options of the following macros now only apply to the current call: \tkzShowLine, \tkzDefSimilitudeCenter, \tkzCalcLength, \tkzGrid, \tkzClip, \tkzDefEquiPoints, \tkzLegend.

– \tkzIsLinear gave a wrong answer (always true); \tkzIsLinear and \tkzIsOrtho now use a tolerance that depends on the lengths.

– The result of \tkzPowerCircle no longer changes after another calculation.

– The inversion of a circle (\tkzDefCircleBy) gives a line only if the circle passes through the center of inversion.

– \tkzDefCircle[spieker] gives the radius in cm, like the other options.

– The Feuerbach point and the Feuerbach triangle, obtained with the option **feuerbach** of \tkzDefTriangleCenter and \tkzDefSpcTriangle, are now exact.

– \tkzDrawPolySeg: the arrow at the start of the path has the right direction.

– \tkzCompass accepts a decimal value for **delta**; \tkzProtractor[return] works; \tkzLabelRegPolygon uses its option **sep**; \tkzDrawX uses its option **step**.

– The old syntax of version 3, for example \tkzDrawCircle[euler](A,B,C), now gives an error message instead of an endless compilation.

– The aliases \tkzDefNinePointsCircle, \tkzFeuerBachCircle and \tkzIncentralTriangle work again.

Two changes may modify your figures:

– \tkzSetUpCompass[delta=...] only acts on the compass marks. To extend the arcs drawn by \tkzDrawArc, \tkzMarkArc and \tkzLabelArc, use \tkzSetUpArc[delta=...]. Before this version, the same value was used for both.

– The Ti*k*Z style **help lines** is no longer modified by the package.

53.3. With version 5.10

– Added a French documentation.

– Added the mini option. You can use this option with the tkz-elements package. Only the modules required for drawing will be loaded. This option is currently only useful if you are using tkz-elements.

53.4. From version 3 to version 5

tkz-euclide 4 was a simplified version of its predecessor, independent of **tkz-base**. Most macros of **tkz-euclide** 3 have been retained, but the special circles are now defined with **\tkzDefCircle** and then drawn with **\tkzDrawCircle**. The unit is now the **cm**. If you need some macros from **tkz-base**, you may need to use **\tkzInit**.
Version 5 does not change anything for the user. With the option `lua` (`\usepackage[lua]{tkz-euclide}`), the calculations are done by Lua: you must then compile with LuaLaTeX. See section 54.

54. Working with lua

If you want Lua to do the calculations, the recommended way is now the package tkz-elements: it computes the points with Lua and **tkz-euclide** draws them (see the option mini below). The option lua of **tkz-euclide** remains available and maintained, but new Lua developments go into tkz-elements.

54.1. Option lua

Since version 5, you can use the option lua of **tkz-euclide**. Just write in your preamble \usepackage[lua]{tkz-euclide}. You must then compile with LuaLaTeX. The syntax does not change.
Without this option, **tkz-euclide** uses the code of version 4, with xfp.
The documentation you are reading has been compiled with this option.
A few words about the method and the results. I considered two possibilities. The first one was simply to replace, wherever I could, the calculations made by xfp with calculations made by lua: this is how I went from fp to xfp and now to lua. The second, more ambitious, would have been to associate a complex number with each point and to compute with complex numbers in lua; this is what tkz-elements now does.
The results are good: this documentation (236 pages at the time) compiled in 47 s with LuaLaTeX and xfp, and in only 30 s with lua.
Another document of 61 pages compiles in 16 s with pdfLaTeX and xfp, and in 13 s with LuaLaTeX and xfp.
This documentation compiles with \usepackage{tkz-base} and \usepackage[lua]{tkz-euclide}, but I did not test all the interactions thoroughly.

54.2. Option mini

When you use tkz-elements to compute the points of your figures, you do not need all the modules of **tkz-euclide**. With the option mini, \usepackage[mini]{tkz-euclide}, only the modules needed for the drawings are loaded. Some macros, such as those of the section *The Show*, still require the complete package.

Index

\ang, 111
\Ax, 175
\Ay, 175

\coordinate, 25

\dAB, 173, 174
\Delta, 162
\draw (A)--(B);, 119

\endpgfinterruptboundingbox, 148
Environment
    scope, 27, 184
    tikzpicture, 184

\foreach, 103
\fpeval, 105

\iftkzFlagCC, 106
\iftkzFlagLC, 101

\len, 174, 175

Operating System
    Windows, 20

Packages
    pgfmath, 238
    tkz-elements, 241, 243
    tkz-euclide, 141
    xfp, 20, 25, 27, 173, 238, 239, 243
\pgfinterruptboundingbox, 148
\pgflinewidth, 116, 117
\pgfmathsetmacro, 105
\pgfresetboundingbox, 141
\px, 175
\py, 175

\rAB, 30
\rAp, 39

standalone, 14

TeX Distributions
    MiKTeX, 20
    TeXLive, 20
Ti*k*Z Library
    babel, 19
    decorations, 238
\tikzset, 183
tkz-euclide: options
    lua, 243
    mini, 243
\tkzAngleResult, 111, 113
\tkzAutoLabelPoint(A,B,C), 159
\tkzAutoLabelPoints, 159
\tkzAutoLabelPoints: arguments
    list of points, 159

\tkzCalcLength, 173, 174, 241
\tkzCalcLength: arguments
(pt1,pt2){name of macro}, 173
\tkzCalcLength: options
cm, 174
\tkzCentroid, 32
\tkzClip, 19, 141, 142, 241
\tkzClip: options
space, 142
\tkzClipBB, 143, 144
\tkzClipCircle[out], 149
\tkzClipCircle, 89, 146
\tkzClipCircle: arguments
(⟨A,B⟩), 146
\tkzClipCircle: options
out, 146
\tkzClipPolygon[out], 145, 149
\tkzClipPolygon, 145
\tkzClipPolygon: arguments
(⟨pt1,pt2,pt3,…⟩), 145
\tkzClipPolygon: options
out, 145
\tkzClipSector(O,A)(B), 147
\tkzClipSector[R](O,2)(30,90), 147
\tkzClipSector[rotate](O,A)(90), 147
\tkzClipSector, 147
\tkzClipSector: options
R, 147
rotate, 147
towards, 147
\tkzcmtopt, 175
\tkzcmtopt: arguments
(number){name of macro}, 175
\tkzCompass, 131, 167, 241
\tkzCompass: options
delta, 167
length, 167
\tkzCompasss, 167, 168
\tkzCompasss: options
delta, 167
length, 167
\tkzDefBarycentricPoint, 32, 34
\tkzDefBarycentricPoint: arguments
(pt1=α_1,pt2=α_2,…), 32
\tkzDefCircle[euler](A,B,C), 238
\tkzDefCircle[R](O,3), 238
\tkzDefCircle[R](O,3cm), 238
\tkzDefCircle[radius](A,B), 174
\tkzDefCircle, 89, 241, 242
\tkzDefCircle: arguments
(⟨pt1,pt2⟩) or (⟨pt1,pt2,pt3⟩), 89
\tkzDefCircle: options
K, 89
R, 89
apollonius, 89
circum, 89
diameter, 89
euler or nine, 89

ex, 89
in, 89
orthogonal from, 89
orthogonal through, 89
spieker, 89
\tkzDefCircleBy, 96, 241
\tkzDefCircleBy: arguments
pt1,pt2, 96
\tkzDefCircleBy: options
homothety, 96
inversion, 96
projection , 96
reflection, 96
rotation , 96
symmetry , 96
translation, 96
\tkzDefEquiPoints, 36, 241
\tkzDefEquiPoints: arguments
(pt1,pt2), 36
\tkzDefEquiPoints: options
/compass/delta, 36
dist, 36
from=pt, 36
show, 36
\tkzDefExtSimilitudeCenter, 33
\tkzDefGoldenRatio(A,C), 31
\tkzDefGoldenRatio, 31
\tkzDefGoldenRatio: arguments
(pt1,pt2), 31
\tkzDefGoldenRectangle, 85
\tkzDefGoldenRectangle: arguments
(⟨pt1,pt2⟩), 85
\tkzDefGoldRectangle, 85
\tkzDefHarmonic, 35
\tkzDefHarmonic: options
both, 35
ext, 35
int, 35
\tkzDefIntSimilitudeCenter, 33
\tkzDefLine, 64
\tkzDefLine: arguments
(⟨pt1,pt2,pt3⟩), 64
(⟨pt1,pt2⟩), 64
(⟨pt1⟩), 64
\tkzDefLine: options
K, 64
altitude, 64
bisector out, 64
bisector, 64
euler, 64
mediator, 64
normed, 64
orthogonal=through…, 64
parallel=through…, 64
perpendicular=through…, 64
symmedian, 64
tangent at, 64
tangent from, 64

\tkzDefMidArc, 36
\tkzDefMidArc: arguments
    pt1, pt2, pt3, 36
\tkzDefMidPoint, 12, 31
\tkzDefMidPoint: arguments
    (pt1,pt2), 31
\tkzDefNinePointsCircle, 241
\tkzDefParallelogram, 28, 84
\tkzDefParallelogram: arguments
    (⟨pt1,pt2,pt3⟩), 84
\tkzDefPoint, 25, 26, 31, 111
\tkzDefPoint: arguments
    (α:d), 26
    (x,y), 26
    {ref}, 26
\tkzDefPoint: options
    label, 26
    shift, 26
\tkzDefPointBy[rotation = …], 111
\tkzDefPointBy, 47
\tkzDefPointBy: arguments
    pt, 47
\tkzDefPointBy: options
    homothety, 47
    inversion negative, 47
    inversion, 47
    projection , 47
    reflection, 47
    rotation in rad, 47
    rotation with nodes, 47
    rotation , 47
    symmetry , 47
    translation, 47
\tkzDefPointOnCircle, 39, 40
\tkzDefPointOnCircle: options
    R in rad, 39
    R, 39
    through in rad, 39
    through, 39
\tkzDefPointOnLine, 38
\tkzDefPointOnLine: arguments
    pt1,pt2, 38
\tkzDefPointOnLine: options
    pos=nb, 38
\tkzDefPoints{0/0/O,2/2/A}, 29
\tkzDefPoints, 25, 29
\tkzDefPoints: arguments
    $x_i/y_i/r_i$, 29
\tkzDefPoints: options
    shift, 29
\tkzDefPointsBy, 47, 55
\tkzDefPointsBy: arguments
    (⟨list of points⟩){⟨list of pts⟩}, 55
\tkzDefPointsBy: options
    homothety = center #1 ratio #2, 55
    inversion = center #1 through #2, 55
    inversion negative = center #1 through #2, 55
    projection = onto #1--#2, 55

reflection = over #1--#2, 55
rotation = center #1 angle #2, 55
rotation in rad = center #1 angle #2, 55
rotation with nodes = center #1 from #2 to #3, 55
symmetry = center #1, 55
translation = from #1 to #2, 55
\tkzDefPointWith, 56
\tkzDefPointWith: arguments
(pt1,pt2), 56
\tkzDefPointWith: options
K, 56
colinear normed= at #1, 56
colinear= at #1, 56
linear normed, 56
linear, 56
orthogonal normed, 56
orthogonal, 56
\tkzDefProjExcenter[name=J](A,B,C)(a,b,c){Y,Z,X}, 93
\tkzDefProjExcenter, 93
\tkzDefProjExcenter: arguments
(pt1=α_1,pt2=α_2,…), 93
\tkzDefProjExcenter: options
name, 93
\tkzDefRadicalAxis, 178
\tkzDefRadicalAxis: arguments
(pt1,pt2)(pt3,pt4), 178
\tkzDefRandPointOn, 62
\tkzDefRandPointOn: options
circle =center pt1 radius dim, 62
circle through=center pt1 through pt2, 62
disk through=center pt1 through pt2, 62
line=pt1--pt2, 62
rectangle=pt1 and pt2, 62
segment= pt1--pt2, 62
\tkzDefRectangle, 84
\tkzDefRectangle: arguments
(⟨pt1,pt2⟩), 84
\tkzDefRegPolygon, 86
\tkzDefRegPolygon: arguments
(⟨pt1,pt2⟩), 86
\tkzDefRegPolygon: options
Options TikZ, 86
center, 86
name, 86
sides, 86
side, 86
\tkzDefShiftPoint, 27, 28
\tkzDefShiftPoint: arguments
(α:d), 28
(x,y), 28
{ref}, 28
\tkzDefShiftPoint: options
[pt], 28
\tkzDefSimilitudeCenter, 33, 241
\tkzDefSimilitudeCenter: arguments
(⟨pt1,pt2⟩)(⟨pt3,pt4⟩), 33
\tkzDefSimilitudeCenter: options
ext, 33

int, 33
\tkzDefSpcTriangle[medial,name=M_](A,B,C){A,B,C}, 75
\tkzDefSpcTriangle[medial,name=M](A,B,C){_A,_B,_C}, 75
\tkzDefSpcTriangle[medial](A,B,C){a,b,c}, 75
\tkzDefSpcTriangle, 75, 241
\tkzDefSpcTriangle: options
centroid or medial, 75
euler, 75
ex or excentral, 75
extouch, 75
feuerbach, 75
in or incentral, 75
intouch or contact, 75
name, 75
orthic, 75
symmedial, 75
tangential, 75
, 75
\tkzDefSquare, 83, 84
\tkzDefSquare: arguments
(⟨pt1,pt2⟩), 83
\tkzDefTriangle, 28, 69, 71
\tkzDefTriangle: options
cheops, 71
egyptian, 71
equilateral, 71
euclid, 71
golden, 71
gold, 71
half, 71
isosceles right, 71
pythagoras, 71
pythagore, 71
school, 71
sublime, 71
swap, 71
two angles= #1 and #2, 71
\tkzDefTriangleCenter[ortho](B,C,A), 41
\tkzDefTriangleCenter, 41, 241
\tkzDefTriangleCenter: arguments
(pt1,pt2,pt3), 41
\tkzDefTriangleCenter: options
centroid, 41
circum, 41
euler, 41
ex, 41
feuerbach, 41
gergonne, 41
grebe, 41
in, 41
lemoine, 41
median, 41
mittenpunkt, 41
nagel, 41
orthic, 41
ortho, 41
spieker, 41
symmedian, 41

\tkzDotProduct, 176
\tkzDotProduct: arguments
    (pt1,pt2,pt3), 176
\tkzDrawArc[angles](O,A)(0,90), 130
\tkzDrawArc[delta=10](O,A)(B), 130
\tkzDrawArc[R with nodes](O,2)(A,B), 130
\tkzDrawArc[R](O,2)(30,90), 130
\tkzDrawArc[rotate,color=red](O,A)(90), 130
\tkzDrawArc, 111, 129, 187, 238, 239, 241
\tkzDrawArc: options
    R with nodes, 130
    R, 130
    angles, 130
    delta, 130
    reverse, 130
    rotate, 130
    towards, 130
\tkzDrawCircle(N,P), 238
\tkzDrawCircle, 89, 124, 135, 242
\tkzDrawCircle: arguments
    (⟨pt1,pt2⟩), 124
\tkzDrawCircles, 125, 238
\tkzDrawCircles: arguments
    (⟨pt1,pt2 pt3,pt4 ...⟩), 125
\tkzDrawCircles: options
    through, 125
\tkzDrawEllipse, 129
\tkzDrawEllipse: arguments
    (⟨C,a,b,An⟩), 129
\tkzDrawLine, 118
\tkzDrawLine: options
    TikZ options, 118
    ..., 118
    add, 118
\tkzDrawLines, 118
\tkzDrawPoint(A,B), 238
\tkzDrawPoint, 116
\tkzDrawPoint: arguments
    name of point, 116
\tkzDrawPoint: options
    TikZ options, 116
    color, 116
    shape, 116
    size, 116
\tkzDrawPoints(A,B), 238
\tkzDrawPoints(A,B,C), 117
\tkzDrawPoints, 116, 117
\tkzDrawPoints: arguments
    points list, 117
\tkzDrawPoints: options
    color, 117
    shape, 117
    size, 117
\tkzDrawPolygon, 122
\tkzDrawPolygon: arguments
    (⟨pt1,pt2,pt3,...⟩), 122
\tkzDrawPolygon: options
    Options TikZ, 122

\tkzDrawPolySeg, 123, 241
\tkzDrawPolySeg: arguments
(⟨pt1,pt2,pt3,...⟩), 123
\tkzDrawPolySeg: options
Options TikZ, 123
\tkzDrawSector(O,A)(B), 133
\tkzDrawSector[R with nodes](O,2)(A,B), 133
\tkzDrawSector[R,color=teal](O,2)(30,90), 133
\tkzDrawSector[rotate,color=red](O,A)(90), 133
\tkzDrawSector, 133–135
\tkzDrawSector: options
R with nodes, 133
R, 133
rotate, 133
towards, 133
\tkzDrawSegment, 118, 119
\tkzDrawSegment: arguments
(pt1,pt2), 119
\tkzDrawSegment: options
TikZ options, 119
..., 119
dim, 119
\tkzDrawSegments, 121, 238
\tkzDrawSemiCircle, 128
\tkzDrawSemiCircle: arguments
(⟨pt1,pt2⟩), 128
\tkzDrawSemiCircles, 128
\tkzDrawSemiCircles: arguments
(⟨pt1,pt2 pt3,pt4 ...⟩), 128
\tkzDrawX, 241
\tkzDuplicateLen, 172
\tkzDuplicateLength, 172
\tkzDuplicateSegment, 172, 173
\tkzDuplicateSegment: arguments
(pt1,pt2)(pt3,pt4){pt5}, 172
\tkzFeuerBachCircle, 241
\tkzFillAngle, 139, 140, 195
\tkzFillAngle: options
size, 140
\tkzFillAngles, 140
\tkzFillCircle, 89, 135
\tkzFillCircle: options
R, 135
radius, 135
\tkzFillPolygon, 138
\tkzFillPolygon: arguments
(⟨pt1,pt2,...⟩), 138
\tkzFillSector(O,A)(B), 139
\tkzFillSector[R with nodes](O,2)(A,B), 139
\tkzFillSector[R,color=blue](O,2)(30,90), 139
\tkzFillSector[rotate,color=red](O,A)(90), 139
\tkzFillSector, 134, 138, 139
\tkzFillSector: options
R with nodes, 139
R, 139
rotate, 139
towards, 139
\tkzFindAngle, 111

\tkzFindAngle: arguments
(pt1,pt2,pt3), 111
\tkzFindSlopeAngle, 113, 114
\tkzFindSlopeAngle: arguments
(pt1,pt2), 113
\tkzGetAngle, 111–113
\tkzGetAngle: arguments
name of macro, 111
\tkzGetFirstPoint{A}, 238
\tkzGetFirstPoint{Jb}, 92
\tkzGetFirstPoint{M} , 30
\tkzGetFirstPoint, 30, 83
\tkzGetFirstPoint: arguments
ref1, 30
\tkzGetLength{dAB}, 173, 174
\tkzGetLength, 30, 89, 173, 174, 238
\tkzGetLength: arguments
name of a macro, 30
\tkzGetPoint(A), 238
\tkzGetPoint{A}, 238
\tkzGetPoint{C}, 56
\tkzGetPoint{M} , 30
\tkzGetPoint{M}, 47
\tkzGetPoint, 12, 30, 31, 41–43, 56, 62, 64, 71, 85, 89, 100, 238
\tkzGetPoint: arguments
ref, 30
\tkzGetPointCoord, 175
\tkzGetPointCoord: arguments
(point){name of macro}, 175
\tkzGetPoints{A}{B}, 238
\tkzGetPoints{M,N} , 30
\tkzGetPoints{N}{P}, 238
\tkzGetPoints{O'}{M'}, 96
\tkzGetPoints{z1}{z2}, 95
\tkzGetPoints, 30, 64, 83–85, 89, 96, 238
\tkzGetPoints: arguments
{ref1,ref2}, 30
\tkzGetRandPointOn, 61
\tkzGetResult, 176
\tkzGetSecondPoint{A}, 238
\tkzGetSecondPoint{M} , 30
\tkzGetSecondPoint{Tb}, 92
\tkzGetSecondPoint, 30, 83
\tkzGetSecondPoint: arguments
ref2, 30
\tkzGetVectxy, 61
\tkzGetVectxy: arguments
(point){name of macro}, 61
\tkzGrid, 241
\tkzIncentralTriangle, 241
\tkzInit, 19, 141, 142, 242
\tkzInit: options
xmax, 142
xmin, 142
xstep, 142
ymax, 142
ymin, 142
ystep, 142

\tkzInterCC, 30, 106
\tkzInterCC: options
    N, 106
    R, 106
    common=pt, 106
    with nodes, 106
\tkzInterLC, 100–102, 241
\tkzInterLC: options
    N, 100
    R, 100
    common=pt, 100
    near, 100
    with nodes, 100
\tkzInterLL, 100
\tkzIsLinear, 180, 181, 241
\tkzIsLinear: arguments
    (pt1,pt2,pt3), 180
\tkzIsOrtho, 180, 181, 241
\tkzIsOrtho: arguments
    (pt1,pt2,pt3), 180
\tkzLabelAngle, 162
\tkzLabelAngle: options
    pos, 162
\tkzLabelAngles, 162, 164
\tkzLabelArc(A,B){5}, 165
\tkzLabelArc, 165, 187, 241
\tkzLabelArc: arguments
    (pt1,pt2,pt3), 165
    label, 165
\tkzLabelArc: options
    pos, 165
\tkzLabelCircle, 89, 164
\tkzLabelCircle: options
    tikz options, 164
\tkzLabelLine(A,B), 162
\tkzLabelLine, 162
\tkzLabelLine: arguments
    label, 162
\tkzLabelLine: options
    pos, 162
\tkzLabelPoint(A){A_1}, 158
\tkzLabelPoint, 158
\tkzLabelPoint: arguments
    point, 158
\tkzLabelPoint: options
    TikZ options, 158
\tkzLabelPoints(A,B,C), 159
\tkzLabelPoints, 159
\tkzLabelPoints: arguments
    list of points, 159
\tkzLabelRegPolygon, 241
\tkzLabelSegment(A,B){5}, 160
\tkzLabelSegment, 160
\tkzLabelSegment: arguments
    (pt1,pt2), 160
    label, 160
\tkzLabelSegment: options
    pos, 160

\tkzLabelSegments, 160, 161
\tkzLegend, 241
\tkzMarkAngle, 153, 154, 195, 238
\tkzMarkAngle: options
    arc, 154
    mark, 154
    mkcolor, 154
    mkpos, 154
    mksize, 154
    size, 154
\tkzMarkAngles, 154
\tkzMarkArc, 153, 187, 241
\tkzMarkArc: options
    color, 153
    mark, 153
    pos, 153
    size, 153
\tkzMarkRightAngle, 155
\tkzMarkRightAngle: options
    german, 155
    size, 155
\tkzMarkRightAngles, 157
\tkzMarkSegment, 152
\tkzMarkSegment: options
    color, 152
    mark, 152
    pos, 152
    size, 152
\tkzMarkSegments, 152
\tkzPermute, 82
\tkzPermute: arguments
    (pt1,pt2,pt3), 82
\tkzPicAngle, 157
\tkzPicAngle: options
    tikz option, 157
\tkzPicRightAngle, 157
\tkzPicRightAngle: options
    tikz option, 157
\tkzPowerCircle, 178, 241
\tkzPowerCircle: arguments
    (pt1)(pt2,pt3), 178
\tkzProtractor, 171, 241
\tkzProtractor: options
    lw, 171
    return, 171
    scale, 171
\tkzpttocm, 174
\tkzpttocm: arguments
    (number){name of macro}, 174
\tkzSetUpArc, 131, 187, 239, 241
\tkzSetUpArc: options
    color, 187
    delta, 187
    line width, 187
    style, 187
\tkzSetUpColors, 183
\tkzSetUpCompass, 187, 188, 239, 241
\tkzSetUpCompass: options

color, 188
delta, 188
line width, 188
style, 188
\tkzSetUpLabel, 188, 189
\tkzSetUpLine, 117, 185
\tkzSetUpLine: options
add, 185
color, 185
line width, 185
style, 185
\tkzSetUpPoint, 183–185
\tkzSetUpPoint: options
color, 183
fill, 183
shape, 183
size, 183
\tkzSetUpStyle, 189
\tkzShowBB, 143
\tkzShowLine, 168, 169, 188, 241
\tkzShowLine: options
K, 168
bisector, 168
gap, 168
length, 168
mediator, 168
orthogonal, 168
perpendicular, 168
ratio, 168
size, 168
\tkzShowTransformation, 169, 170
\tkzShowTransformation: options
K, 170
gap, 170
length, 170
projection=onto pt1--pt2, 170
ratio, 170
reflection= over pt1--pt2, 170
size, 170
symmetry=center pt, 170
translation=from pt1 to pt2, 170
\tkzSwapPoints, 176
\tkzSwapPoints: arguments
(pt1,pt2), 176
\tkzTestInterCC, 100, 106, 107
\tkzTestInterLC, 100, 101

\useasboundingbox, 141

\Vx, 61
\Vy, 61

\xstep, 142

\ystep, 142