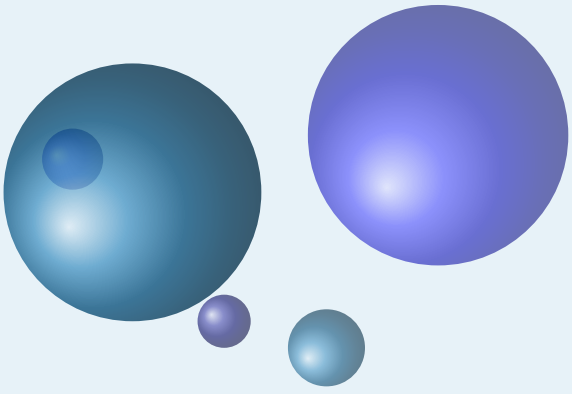
tkz-euclide 3.01c

AlterMundus



Alain Matthes

23 janvier 2020 Documentation V.3.01c

http://altermundus.fr

tkz-euclide

AlterMundus

Alain Matthes

☞ Le package **tkz-euclide** est un ensemble de macros spécialisées permettant de construire des figures de géométrie euclidienne dans un plan muni d'un repère. Le package est construit au-dessus de PGF et de son interface TikZ. Ce document fournit les définitions des différentes macros ainsi que des exemples. **tkz-euclide** nécessite une version supérieure ou égale à 3.0 de Ti*k*Z. **tkz-euclide** est avant tout destiné aux professeurs de mathématiques, à leurs étudiants et élèves.

☞ Je souhaite remercier **Till Tantau** pour avoir créé le merveilleux outil Ti*k*Z, ainsi que **Michel Bovani** pour **fourier**.

☞ Je remercie **Yve Combe** pour avoir partagé son travail sur le rapporteur et les constructions à l'aide du compas. Je souhaite remercier également, **David Arnold** qui a corrigé un grand nombre d'erreurs et qui a testé de nombreux exemples, **Wolfgang Büchel** qui a corrigé également des erreurs et a construit de superbes scripts pour obtenir les fichiers d'exemples, **John Kitzmiller** et **Dimitri Kapetas** pour leurs exemples, et enfin **Gaétan Marris** pour ses remarques et corrections.

☞ Vous trouverez d'autres exemples sur mon site : altermundus.fr

Vous pouvez envoyer vos remarques, et les rapports sur des erreurs que vous aurez constatées à l'adresse suivante : Alain Matthes.

Table des matières

1 Nouveautés et présentation

Le package **tkz-euclide** utilise un repère cartésien normé (unité le cm ou une unité équivalente) pour tracer pas-à-pas des figures de géométrie euclidienne. Certaines modifications ont été apportées afin de rendre plus homogène la syntaxe et surtout afin de bien distinguer la définition et la recherche de coordonnées du reste c'est-à-dire dessin, marquage et étiquetage. Dans le futur, les macros de définition étant isolées, il sera plus facile d'introduire une phase de calculs de coordonnées à l'aide de **Lua**

Une nouveauté importante est le remplacement récent du package **fp** par **xfp**. Il s'agit d'améliorer un peu plus les calculs et de faciliter l'utilisation.

Voici quelques unes des modifications.

- Amélioration du code et correction de bugs
- Avec **tkz-euclide** charge tous les objets, donc plus besoin de placer **\usetkzobjall**.
- La "bounding box" est désormais contrôlée dans chaque macro (enfin je l'espère) cela permet d'éviter l'utilisation de **\tkzInit** suivi de **\tkzClip**
- Ajout de macros concernant la "bounding box" : **\tkzSaveBB** **\tkzClipBB** etc.
- Logiquement la plupart des macros acceptent les options de TikZ. J'ai donc retiré les options "doublons" lorsque c'était possible ; ainsi l'option "label options" est supprimée
- Les points aléatoires sont désormais dans **tkz-euclide** et la macro **\tkzGetRandPointOn** est remplacée par **\tkzDefRandPointOn**. Pour des raisons d'homogénéité, il faut récupérer les points avec **\tkzGetPoint**.
- Les options **end** et **start** qui permettaient de donner un label à une droite sont supprimées. Il faut désormais utiliser la macro **\tkzLabelLine**
- Introduction des librairies `quotes` et `angles` cela permet de donner un label à un point.même si je ne suis pas favorable à cette pratique.
- La notion de vecteur disparait pour tracer un vecteur il suffit de passer "->" en option de **\tkzDrawSegment**.
- De nombreuses macros existent encore, mais sont obsolètes et disparaitront :
  - `\tkzDrawMedians` trace et créer des points milieux des côtés d'un triangle. La séparation création et dessin n'est pas respectée aussi il est préférable d'abord de créer les coordonnées de ces points par `\tkzSpcTriangle[median]` puis de choisir celles que l'on va tracer avec `\tkzDrawSegments` ou encore `\tkzDrawLines`
  - Autre exemple `\tkzDrawTriangle[equilateral]` était pratique mais il est préférable d'obtenir le troisième point par `\tkzDefTriangle[equilateral]` puis de tracer avec `\tkzDrawPolygon`.
  - Enfin `\tkzDrawCircle`
  - `\tkzDefRandPointOn` remplac par `\tkzGetRandPointOn`

  `\tkzDraw`
- Apparition de la macro **\usetkztool** qui permet de charger de nouveaux "outils".

2 Installation

tkz-euclide et **tkz-base** sont désormais sur le serveur du **CTAN**[1]. Si vous voulez tester une version beta, il vous suffit de placer les fichiers suivants dans un dossier texmf que votre système pourra trouver. Il vous faudra vérifier plusieurs points :

— Les dossiers **tkz-base** et **tkz-euclide** doivent être situés sur un chemin reconnu par **latex**.

— **xfp**[2], **numprint** , **tikz 3.00** doivent être installés car ils sont obligatoires, pour le bon fonctionnement de **tkz-euclide**.

— Cette documentation ainsi que tous les exemples ont été obtenus avec **lualatex-dev** mais **pdflatex** devrait convenir.

2.1 Liste des fichiers des dossiers tkzbase et tkzeuclide

Dans le dossier **base** :

— **tkz-base.cfg**
— **tkz-base.sty**
— **tkz-lib-marks.tex**
— **tkz-obj-axes.tex**
— **tkz-obj-circles.tex**
— **tkz-obj-grids.tex**
— **tkz-obj-lines.tex**
— **tkz-obj-marks.tex**
— **tkz-obj-points.tex**
— **tkz-obj-rep.tex**
— **tkz-obj-segments.tex**
— **tkz-tools-arith.tex**
— **tkz-tools-base.tex**
— **tkz-tools-BB.tex**
— **tkz-tools-math.tex**
— **tkz-tools-misc.tex**
— **tkz-tools-modules.tex**
— **tkz-tools-print.tex**
— **tkz-tools-utilities.tex**

Dans le dossier **euclide** :

— **tkz-euclide.sty**
— **tkz-obj-angles.tex**
— **tkz-obj-arcs.tex**
— **tkz-obj-compass.tex**
— **tkz-obj-defcircles.tex**
— **tkz-obj-deflines.tex**
— **tkz-obj-defpoints.tex**
— **tkz-obj-defpointsby.tex**
— **tkz-obj-defpointsrnd.tex**
— **tkz-obj-defpointswith.tex**
— **tkz-obj-polygons.tex**
— **tkz-obj-protractor.tex**
— **tkz-obj-sectors.tex**
— **tkz-obj-show.tex**

1. tkz-base et tkz-euclide font partie de TeXLive et tlmgr permet de les installer. Ces packages font aussi partie de MikTeX sous Windows
2. xfp remplace fp

— tkz-obj-triangles.tex
— tkz-tools-intersections.tex

Désormais **tkz-euclide** charge tous les fichiers.

3 Présentation

3.1 À propos de TikZ et que peut apporter tkz-euclide ?

TikZ est un outil que je trouve très agréable à utiliser. J'ai trouvé si simple son utilisation que je me suis demandé si cela avait un sens de créer un package pour la création de dessins géométriques. Quels arguments peuvent intervenir ?

1. Certains utilisateurs n'ont pas envie d'apprendre quoi que ce soit sur Ti*k*Z, cela est respectable et une simplification du code par l'intermédiaire d'un package peut avoir une certaine utilité. La syntaxe n'est plus tout à fait celle de Ti*k*Z, mais ressemble davantage à celle de LaTeX.
2. Les noms des macros ont une *signification plus mathématique*.
3. La grande différence avec Ti*k*Z est qu'il est possible d'utiliser de grandes valeurs ainsi que des très petites, car la majorité des calculs sont faits à l'aide de **xfp**. C'est plus lent, mais plus précis.
4. Il est possible de modifier facilement les styles pour les objets principaux que sont les points, les droites, les cercles, les arcs, etc. cela se fait cette fois comme avec Ti*k*Z.
5. Des exemples de constructions géométriques sont fournis et peuvent être utiles au débutant.
6. Et pour terminer, cela peut être une approche en douceur de l'utilisation de Ti*k*Z par l'intermédiaire des options. Dans cette nouvelle version, j'ai essayé que les options de Ti*k*Z soient pratiquement toujours disponibles.

Je vous encourage toutefois à étudier Ti*k*Z. Je donnerai quelques exemples pour voir les différences entre les codes. Cela dit, il est toujours possible de mélanger les différents codes et différentes syntaxes, cela n'est pas franchement satisfaisant, mais peut permettre de résoudre certains problèmes.

3.2 À propos de tkz-euclide

Le but est donc de créer des figures de géométrie euclidienne, pas-à-pas à l'aide de macros ayant un but unique une figure, en suivant la méthode de dessin à la main. Avec **tkz-euclide**, l'unité par défaut est le centimètre. Si votre travail ne concerne que de la géométrie classique, je vous conseille de conserver cette unité.

4 Syntaxe

Quelques mots sur la syntaxe.

☞ Les commandes de **tkz-euclide** sont des macros comme celles de LaTeX. Leurs noms comment par **tkz**. Les macros utilisent des signes traditionnels qui sont : les crochets, les parenthèses et les accolades. Les *accolades* sont réservées pour la création d'objets et les *parenthèses* ne sont utilisées que pour des objets, déjà existants, quant aux crochets, ils sont réservés aux options :

\tkzDefPoint(1,2){A} crée le point nommé A.

\tkzLabelSegment[below](O,A){1} crée le label 1 pour le segment $[OA]$.

Enfin des macros comme **\tkzDefMidPoint(O,A)** créent un point, qui est ici, le milieu d'un segment. Le point est nommé temporairement **tkzPointResult**.

Soit la création est une étape intermédiaire, et vous n'avez pas besoin de conserver ce point, alors tant qu'aucune macro ne modifie l'attribution de **tkzPointResult**, vous pouvez utiliser ce nom pour faire référence au milieu ; soit vous voulez conserver ce point, car il sera utilisé plusieurs fois, alors la macro **\tkzGetPoint{M}** permet d'attribuer le nom **M** au point.

Quant une macro donne comme résultat deux points, le premier est nommé **tkzFirstPointResult** et le second **tkzSecondPointResult**, la macro qui permet de récupérer les points est :

☞ L'ordre des points n'est pas prédéfini !

— **\tkzGetPoints{M}{N}** qui attribue deux noms ;
— **\tkzGetFirstPoint{M}** seul le premier point sera utilisé ;
— **\tkzGetSecondPoint{N}** cette fois, seul le second point est nommé.

Il est difficile de conserver un découpage du code comme dans l'exemple, si on ne veut pas nommer un point par exemple H dans l'exemple minimal, mais complet de la section suivante.

Le code pourrait devenir :

```
\tkzDefPointWith[orthogonal](I,M) %\tkzGetPoint{H}
\tkzDrawSegment[style=dashed](I,tkzPointResult)
\tkzInterLC(I,tkzPointResult)(M,A)    \tkzGetSecondPoint{B}
```

4.1 Notions générales : création d'une figure

On utilise l'environnement **tikzpicture** de Ti*k*Z . Le principe est de définir des points "fixes" en utilisant des coordonnées cartésiennes ou des coordonnées polaires.

Ensuite, il est possible de définir des lignes, des cercles puis d'obtenir d'autres points comme intersections d'objets, comme images d'autres points à l'aide de transformations ou bien encore des points issus de propriétés vectorielles.

— **\tkzDefPoint** pour l'usage de coordonnées,
— **\tkzDefPointBy** pour l'usage des transformations,
— **\tkzDefPointWith** pour l'usage des propriétés vectorielles,
— et enfin **\tkzInterLL**, **\tkzInterLC** et **\tkzInterCC** sont les trois types d'intersections possibles de droites et de cercles. Pour ces trois macros, j'ai préféré utiliser **xfp** afin d'obtenir des résultats plus précis.

Puis à l'aide de ces points, nous pouvons tracer des objets comme des segments, des demi-droites, des droites, des triangles, des cercles, des arcs etc.

Cela se fait à l'aide de macros dont le nom commence par **\tkzDraw...**.

Enfin il est possible de placer des labels à l'aide de macros dont le nom commence par **\tkzLabel...**.

Cela permet à ceux qui le souhaitent, de décomposer la création des figures en six étapes :

1. *Initialisation* (*préparation repère, unités*)
2. *Définir les points dont les coordonnées sont connues ou bien calculables.*
3. *Création de nouveaux points à l'aide de méthodes* (*intersection, transformation,etc.*).
4. *Tracés des objets dans un ordre choisi.*
5. *Marquage graphique.*
6. *Placement des labels.*

Pourquoi ?

a) Les trois premières parties peuvent utiliser d'autres outils que Ti*k*Z comme **fp**, **xfp** ou encore **lua**. Par exemple pour des tracés de courbes, Ti*k*Z peut faire appel à **gnuplot**. Si dans la majorité des cas Ti*k*Z est suffisant, pour certains calculs complexes et il est préférable d'utiliser des outils plus adaptés. Les coordonnées peuvent être obtenues à l'aide de calculs en utilisant **pgfmath** , **xfp** ou encore TEX. Toutes les macros n'acceptent pas que les calculs soient faits pendant leurs assignations. **xfp** est plus précis que **pgfmath**, plus rapide aussi tout dépend des constructions demandées. Il est également envisagé d'introduire **lua** afin d'obtenir une meilleure précision.

b) Pour les trois dernières parties, Ti*k*Z est parfait. **tkz-euclide** ne sert qu'à donner une syntaxe plus mathématicienne.

D'une façon générale, la syntaxe est plus homogène. Les noms des points créés sont entre accolades alors que les noms des points utilisés sont entre parenthèses.

Après beaucoup d'hésitations, j'ai choisi le procédé suivant. Quand une macro crée un point, deux points alors le résultat est rangé dans un nom de générique. Ainsi l'intersection de deux droites définit un point

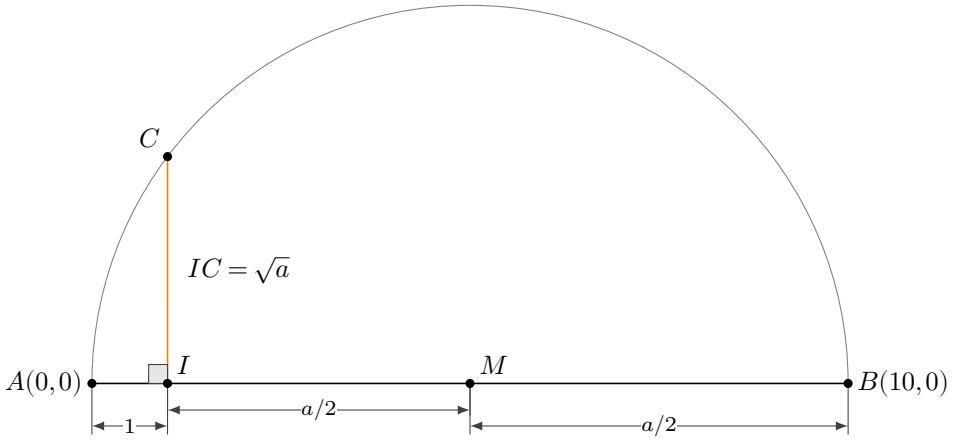
appelé **tkzPointResult**, celle de deux cercles donne **tkzFirstPointResult** et **tkzSecondPointResult**. Certaines macros définissent une mesure de rayon qui sera alors dans une macro **\tkzLengthResult** et d'autres la mesure d'un angle **\tkzAngleResult**.

1. La première contrainte est celle de la taille de la figure que l'on veut créer. Désormais l'initialisation à l'aide la macro **\tkzInit** n'est plus nécessaire, à l'exception de cas complexes où il faudra "clipper" une partie de la figure ;
2. Il faut remarquer que l'unité choisie pour **tkz-euclide** est le cm ou toute autre unité équivalente. Il est préférable soit de modifier les coordonnées soit d'utiliser l'option **scale** ;
3. Ensuite il s'agit de donner les points de base (points fixes) à partir desquels la construction se fera ;
4. Enfin des points seront obtenus à l'aide de différentes transformations.

5 Exemple minimal, mais complet

Une unité de longueur étant choisie, l'exemple montre comment obtenir un segment de longueur $\sqrt{a}$ à partir d'un segment de longueur a, à l'aide d'une règle et d'un compas.

$IB = a$, $AI = 1$



Commentaires

— Le péambule
Voyons tout d'abord le préambule. Si vous en avez besoin, il faut charger **xcolor** avant **tkz-euclide** c'est-à-dire avant Ti*k*Z . Ti*k*Z peut poser des problèmes avec les caractères actifs de **frenchb** de **babel** mais fournit dans sa dernière version une librairie censée résoudre ces problèmes `babel`.

```
\documentclass{standalone} % ou une autre classe
   % \usepackage{xcolor} % avant tikz ou tkz-euclide si nécessaire
\usepackage{tkz-euclide} % no need to load TikZ
   % \usetkzobj{all}  n'est plus nécessaire
   % \usetikzlibrary{babel}  si les caratères actifs posent des problèmes
```

Le code suivant comprend plusieurs parties :

— Définition des points fixes : la première partie comprend les définitions de points nécessaires à la contruction, ce sont les points fixes. Les macros **\tkkzInit** et **\tkkzClip** dans la plupart des cas ne sont nécessaires.

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

```
\tkzDefPoint(10,0){B}
```

— La deuxième partie est dédiée à la création de nouveaux points à partir des points fixes ; un point B est placé à $10cm$ de A. On définit le milieu de $[AB]$ par M puis on recherche la droite orthogonale à la droite (AB) au point I. Ensuite, on cherche l'intersection de cette droite avec le demi-cercle de centre M passant par 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,A)
   \tkzGetSecondPoint{B}
```

— La troisième comprend les différents tracés ;

```
\tkzDrawSegment[style=dashed](I,H)
\tkzDrawPoints(O,I,A,B,M)
\tkzDrawArc(M,A)(O)
\tkzDrawSegment[dim={$1$,-16pt,}](O,I)   % voir la documentation pour l'usage de dim
\tkzDrawSegment[dim={$a/2$,-10pt,}](I,M)
\tkzDrawSegment[dim={$a/2$,-16pt,}](M,A)
```

— Marquage : la quatrième est consacrée au marquage ;

```
\tkzMarkRightAngle(A,I,B)
```

— Étiquetage : la dernière ne s'occupe que du placement des labels.

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

— Le code complet :

```
\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{i}
   \tkzInterLC(I,i)(M,B)             \tkzGetSecondPoint{C}

   \tkzDrawSegment[style=orange](I,C)
   \tkzDrawArc(M,B)(A)
   \tkzDrawSegment[dim={$1$,-16pt,}](A,I)
   \tkzDrawSegment[dim={$a/2$,-10pt,}](I,M)
   \tkzDrawSegment[dim={$a/2$,-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){$IC=\sqrt{a}$}
\end{tikzpicture}
```

6 Résumé de tkz-base

6.1 Utilité de tkz-base

Premièrement, il faut savoir qu'il n'est pas nécessaire de s'occuper avec Ti*k*Z de la taille du support "bounding box". Les premières versions de **tkz-euclide** ne contrôlaient pas l'extension de la "bounding box", désormais l'extension de cette box est limitée.

Cependant, il est parfois nécessaire de contrôler la taille de ce qui sera affiché. Pour cela, il faut avoir préparé le repère dans lequel vous allez travailler, c'est le rôle de **tkz-base** et de sa macro principale **\tkzInit**. Il est recommandé de laisser l'unité graphique égale à 1 cm. Pour certains dessins, il est intéressant de fixer les valeurs extrêmes (xmin,xmax,ymin et ymax) et de « clipper » le rectangle de définition afin de contrôler au mieux la taille de la figure.

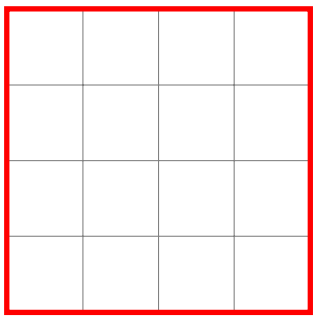
Les deux macros de **tkz-base** utiles pour **tkz-euclide** sont :

— **\tkzInit**
— **\tkzClip**

À cela, j'ai ajouté des macros liées directement à la "bounding box". Vous pouvez désormais la visualiser, la sauvegarder, la restaurer (voir la documentation de **tkz-base** section BB)

6.2 \tkzInit et \tkzShowBB

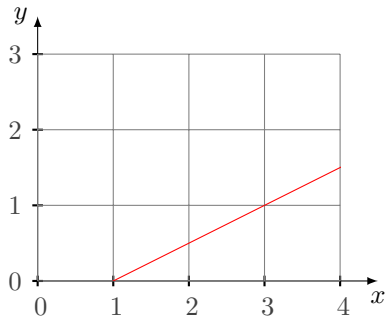
Le rectangle autour de la figure vous montre la "bounding box"



```
\begin{tikzpicture}
 \tkzInit[xmin=-1,xmax=3,ymin=-1, ymax=3]
 \tkzGrid
 \tkzShowBB[red,line width=2pt]
\end{tikzpicture}
```

6.3 \tkzClip

Le rôle de cette macro est de « clipper » le rectangle initial afin que seuls les tracés contenus dans ce rectangle soient dessinés.



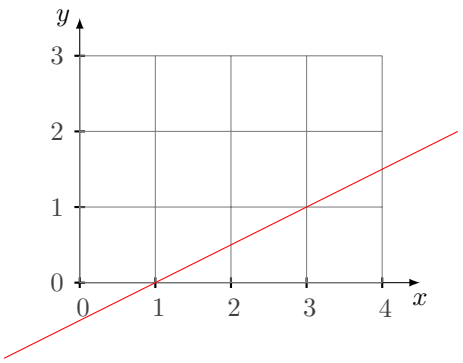

```
\begin{tikzpicture}
 \tkzInit[xmax=4, ymax=3]
 \tkzAxeXY
 \tkzGrid
 \tkzClip
 \draw[red] (-1,-1)--(5,2);
\end{tikzpicture}
```

Il est possible d'ajouter un peu d'espace

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

6.4 \tkzClip et l'option space

Cette option permet d'ajouter un peu d'espace autour du rectangle "clippé".




```
\begin{tikzpicture}
 \tkzInit[xmax=4, ymax=3]
 \tkzAxeXY
 \tkzGrid
 \tkzClip[space=1]
 \draw[red] (-1,-1)--(5,2);
\end{tikzpicture}
```

les dimensions du rectangle "clippé" sont **xmin-1**, **ymin-1**, **xmax+1** et **ymax+1**.

7 Définition d'un point

La macro **\tkzDefPoint** permet de définir un point en lui attribuant des coordonnées. Cette macro est basée sur **\coordinate**, macro de Ti*k*Z . Elle peut utiliser des options propres à Ti*k*Z comme `shift`. Si des calculs sont nécessaires alors c'est le package **xfp** qui est choisi. On peut utiliser les coordonnées cartésiennes ou polaires.

7.1 Définition d'un point en coordonnées cartésiennes : \tkzDefPoint

\tkzDefPoint[⟨local options⟩](⟨x,y⟩){⟨name⟩} ou (⟨a:r⟩){⟨name⟩}

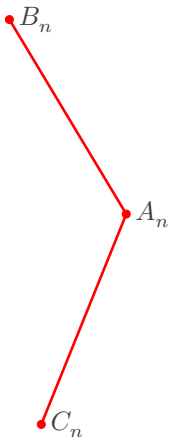
arguments	défaut	définition
(x,y)	no default	x et y sont deux dimensions, par défaut en cm.
(a:d)	no default	a est un angle en degré, d une dimension
{name}	no default	Nom attribué au point : A, T_a ,I_1 etc ...

Les arguments obligatoires de cette macro sont deux dimensions exprimées avec des décimaux, dans le premier cas ce sont deux mesures de longueur, dans le second ce sont une mesure de longueur et la mesure d'un angle en degré

options	défaut	définition
label	no default	permet de placer un label à une distance prédéfinie
shift	no default	Ajoute (x,y) ou (a:d) à toutes les coordonnées

7.1.1 scope pour une combinaison et shift

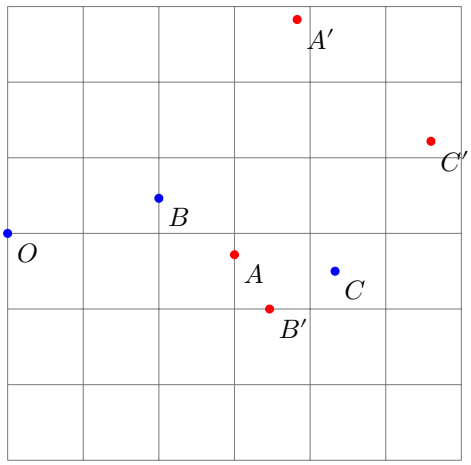
Préférable pour effectuer une rotation, est d'utiliser un environnement **scope**. J'ai utilisé ici l'option `label` mais cela est à éviter.




```
\begin{tikzpicture}[rotate=90]
 \tkzDefPoint[label=right:$A_n$](2,3){A}
 \begin{scope}[shift={(A)}]
  \tkzDefPoint[label= right:$B_n$](31:3){B}
  \tkzDefPoint[label= right:$C_n$](158:3){C}
 \end{scope}
  \tkzDrawSegments[color=red,%
           line width=1pt](A,B A,C)
  \tkzDrawPoints[color=red](A,B,C)
 \end{tikzpicture}
```

7.1.2 Formules et coordonnées

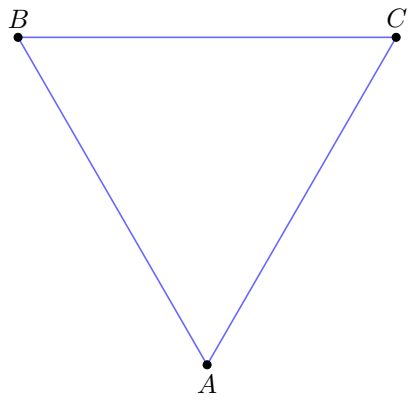
Il faut ici respecter la syntaxe de **fxp**. Il est toujours possible de passer par **pgfmath** mais dans ce cas, il faut calculer les coordonnées avant d'utiliser la macro **\tkzDefPoint**.




```
\begin{tikzpicture}[scale=1]
  \tkzInit[xmax=6,ymax=6]
  \tkzGrid
  \tkzDefPoint(-1+1,-1+4){O}
  \tkzDefPoint({3*ln(exp(1))},{exp(1)}){A}
  \tkzDefPoint({4*sin(pi/6)},{4*cos(pi/6)}){B}
  \tkzDefPoint({4*sin(pi/3)},{4*cos(pi/3)}){B'}
  \tkzDefPoint(30:5){C}
  \tkzDefPoint[shift={(1,3)}](45:4){A'}
  \begin{scope}[shift=(A)]
      \tkzDefPoint(30:3){C'}
  \end{scope}
  \tkzDrawPoints[color=blue](O,B,C)
  \tkzDrawPoints[color=red](B',A,A',C')
  \tkzLabelPoints(A,O,B,B',A',C,C')
\end{tikzpicture}
```

7.1.3 Scope et \tkzDefPoint

On peut tout d'abord utiliser l'environnement **scope** de Ti*k*Z . Dans l'exemple suivant, nous avons un moyen de définir un triangle équilatéral.




```
\begin{tikzpicture}[scale=1]
  \tkzSetUpLine[color=blue!60]
 \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}
```

7.2 Définition de points multiples : \tkzDefPoints

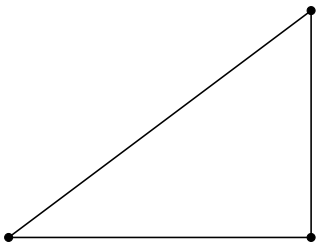
\tkzDefPoints[⟨local options⟩]{⟨$x_1/y_1/n_1, x_2/y_2/n_2$, ...⟩}

x_i et y_i sont les coordonnées d'un point référencé n_i

arguments	exemple
$x_i/y_i/n_i$	**\tkzDefPoints{0/0/O,2/2/A}**

options	défaut	définition
label	no default	permet de placer un label à une distance prédéfinie
shift	no default	Ajoute (x,y) ou (a:d) à toutes les coordonnées

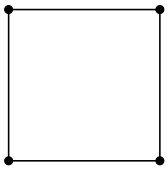
7.3 Créer un triangle avec \tkzDefPoints



```
\begin{tikzpicture}[scale=1]
 \tkzDefPoints{0/0/A,4/0/B,4/3/C}
 \tkzDrawPolygon(A,B,C)
 \tkzDrawPoints(A,B,C)
\end{tikzpicture}
```

7.4 Créer un carré avec \tkzDefPoints

Voir définir un carré 20.1. Remarquez ici la syntaxe pour dessiner le polygone.



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

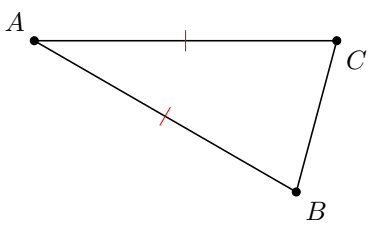
7.5 Point relativement à un autre : \tkzDefShiftPoint

\tkzDefShiftPoint[⟨Point⟩](⟨x,y⟩){⟨name⟩} ou (⟨a:r⟩){⟨name⟩}

arguments	défaut	définition
(x,y)	no default	x et y sont deux dimensions, par défaut en cm.
(a:r)	no default	a est un angle en degré, r une dimension
options	**défaut**	**définition**
[pt]	no default	\tkzDefShiftPoint[A](0:4){B}

7.5.1 Triangle isocèle avec \tkzDefShiftPoint

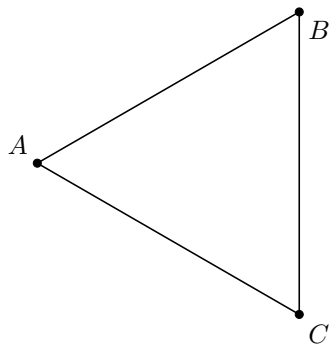
Cette macro permet de placer un point relativement à un autre. Cela revient à une translation. Voici comment construire un triangle isocèle de sommet principal A et d'angle au sommet de 30 degrés.




```
\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=|,
           color=red](A,B A,C)
 \tkzDrawPoints(A,B,C)
 \tkzLabelPoints(B,C)
 \tkzLabelPoints[above left](A)
\end{tikzpicture}
```

7.5.2 Triangle équilatéral avec \tkzDefShiftPoint

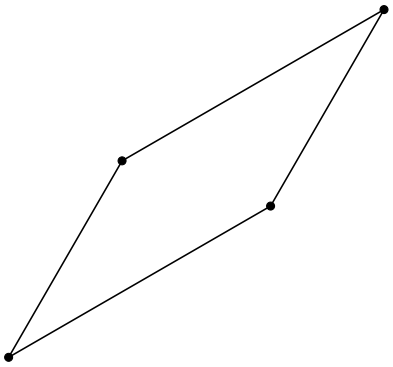
Voyons comment obtenir un triangle équilatéral (il y a beaucoup plus simple) 18.1.2




```
\begin{tikzpicture}[scale=1]
 \tkzDefPoint(2,3){A}
 \tkzDefShiftPoint[A](30:4){B}
 \tkzDefShiftPoint[A](-30:4){C}
 \tkzDrawPolygon(A,B,C)
 \tkzDrawPoints(A,B,C)
 \tkzLabelPoints(B,C)
 \tkzLabelPoints[above left](A)
\end{tikzpicture}
```

7.5.3 Parallélogramme avec \tkzDefShiftPoint

Il y a plus simple



```
\begin{tikzpicture}
 \tkzDefPoint(0,0){A}
 \tkzDefPoint(60:3){B}
 \tkzDefShiftPointCoord[B](30:4){C}
 \tkzDrawPolygon(A,...,D)
 \tkzDrawPoints(A,...,D)
\end{tikzpicture}
```

8 Points particuliers

L'introduction des points a été réalisée dans **tkz-base**, la macro la plus importante étant **\tkzDefPoint**. Voici quelques points particuliers.

8.1 Milieu d'un segment \tkzDefMidPoint

Il s'agit de déterminer le milieu d'un segment.

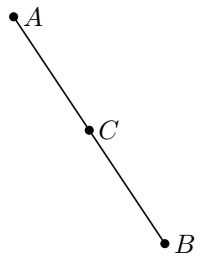
\tkzDefMidPoint(⟨pt1,pt2⟩)

Le résultat est dans **tkzPointResult**. On peut le récupérer avec **\tkzGetPoint**.

arguments	défaut	définition
(pt1,pt2)	no default	pt1 et pt2 sont deux points

8.1.1 Utilisation de \tkzDefMidPoint

Revoir l'utilisation de **\tkzDefPoint** dans .




```
\begin{tikzpicture}[scale=1]
 \tkzDefPoint(2,3){A}
 \tkzDefPoint(4,0){B}
 \tkzDefMidPoint(A,B) \tkzGetPoint{C}
 \tkzDrawSegment(A,B)
 \tkzDrawPoints(A,B,C)
 \tkzLabelPoints[right](A,B,C)
\end{tikzpicture}
```

8.2 Coordonnées barycentriques \tkzDefBarycentricPoint

$pt_1, pt_2, ..., pt_n$ étant n points, ils définissent n vecteurs $\overrightarrow{v_1}, \overrightarrow{v_2}, ..., \overrightarrow{v_n}$ avec comme extrémité commune l'origine du repère. $\alpha_1, \alpha_2, ..., \alpha_n$ étant n nombres, le vecteur obtenu par :

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

définit un point unique.

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

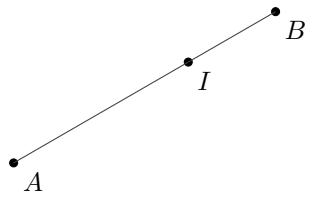
arguments	défaut	définition
(pt1=α_1,pt2=α_2,…)	no default	Chaque point a une pondération

Il faut au moins deux points.

8.2.1 Utilisation de \tkzDefBarycentricPoint avec deux points

Nous obtenons dans l'exemple suivant le barycentre des points A et B affectés des coefficients 1 et 2, autrement dit :

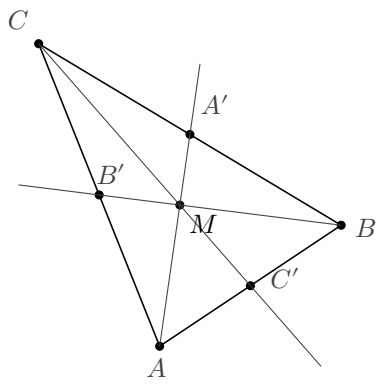
$$\overrightarrow{AI}=\frac{2}{3}\overrightarrow{AB}$$




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

8.2.2 Utilisation de \tkzDefBarycentricPoint avec trois points

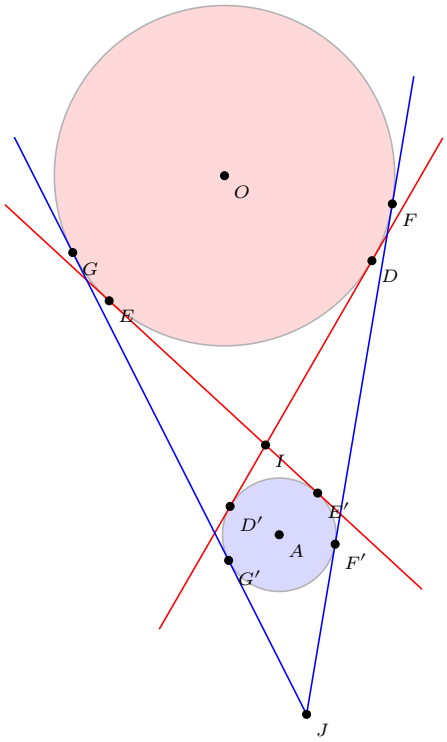
Cette fois M est simplement le centre de gravité du triangle. Pour des raisons de simplification et d'homogénéité, il existe aussi **\tkzCentroid**




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

8.3 Centre de similitude interne \tkzDefIntSimilitudeCenter et \tkzDefExtSimilitudeCenter

On nomme centre de similitude externe et centre de similitude interne, les centres des deux homothéties dans lesquelles se correspondent deux cercles.




```
\begin{tikzpicture}[scale=.75,rotate=-30]
 \tkzDefPoint(0,0){O}
 \tkzDefPoint(4,-5){A}
 \tkzDefIntSimilitudeCenter(O,3)(A,1) \tkzGetPoint{I}
 \tkzExtSimilitudeCenter(O,3)(A,1) \tkzGetPoint{J}
 \tkzDefTangent[from with R= I](O,3 cm)  \tkzGetPoints{D}{E}
 \tkzDefTangent[from with R= I](A,1 cm)  \tkzGetPoints{D'}{E'}
 \tkzDefTangent[from  with R= J](O,3 cm) \tkzGetPoints{F}{G}
 \tkzDefTangent[from with R= J](A,1 cm)   \tkzGetPoints{F'}{G'}
 \tkzDrawCircle[R,fill=red!50,opacity=.3](O,3 cm)
 \tkzDrawCircle[R,fill=blue!50,opacity=.3](A,1 cm)
 \tkzDrawSegments[add = .5 and .5,color=red](D,D' E,E')
 \tkzDrawSegments[add= 0 and 0.25,color=blue](J,F J,G)
 \tkzDrawPoints(O,A,I,J,D,E,F,G,D',E',F',G')
 \tkzLabelPoints[font=\scriptsize](O,A,I,J,D,E,F,G,D',E',F',G')
\end{tikzpicture}
```

9 Points particuliers relatifs à un triangle

9.1 Centre de triangle : \tkzDefTriangleCenter

Cette macro permet de définir le centre d'un triangle.

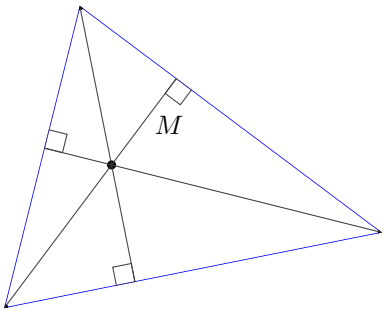
\tkzDefTriangleCenter[⟨local options⟩](⟨A,B,C⟩)

Attention les arguments sont des listes de trois points. Cette macro est utilisée en partenariat avec **\tkzGetPoint** pour obtenir le centre cherché. On peut utiliser **tkzPointResult** s'il n'est pas nécessaire de conserver les résultats.

arguments	défaut	définition
(pt1,pt2,pt3)	no default	trois points
options	**défaut**	**définition**
ortho	circum	Interesction des hauteurs d'un triangle
centroid	circum	centre de gravité. Intersection des médianes
circum	circum	centre du cercle circonscrit
in	circum	centre du cercle inscrit dans à un triangle
ex	circum	centre d'un cercle exinscrit à un triangle
euler	circum	centre du ercle d'Euler
symmedian	circum	Le point de Lemoine ou centre symmedian ou point de Grebe
spieker	circum	centre du Cercle de Spieker
nagel	circum	Centre de Nagel
mittenpunkt	circum	ou encore MiddlePoint center
feuerbach	circum	Point de Feuerbach

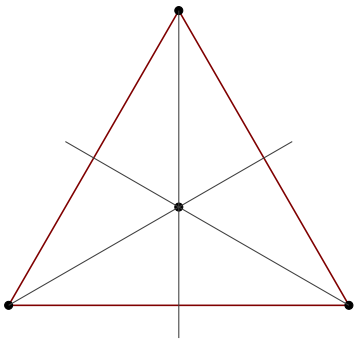
9.1.1 Orthocentre \tkzDefTriangleCenter[ortho] ou \tkzDefTriangleCenter[orthic]

Les trois hauteurs d'un triangle sont concourantes. Leur point d'intersection H, est nommé orthocentre du triangle.



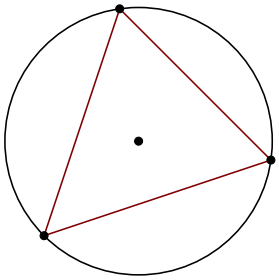

```
\begin{tikzpicture}
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(5,1){B}
  \tkzDefPoint(1,4){C}
  \tkzClipPolygon(A,B,C)
  \tkzDefTriangleCenter[ortho](B,C,A)
    \tkzGetPoint{H}
  \tkzDefSpcTriangle[orthic,name=H](A,B,C){a,b,c}
  \tkzDrawPolygon[color=blue](A,B,C)
  \tkzDrawPoints(A,B,C,H)
  \tkzDrawLines[add=0 and 1](A,Ha B,Hb C,Hc)
  \tkzLabelPoint(M){$M$}
  \tkzAutoLabelPoints[center=H](A,B,C)
  \tkzMarkRightAngles(A,Ha,B B,Hb,C C,Hc,A)
\end{tikzpicture}
```

9.1.2 Centre de gravité \tkzDefTriangleCenter[centroid]



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

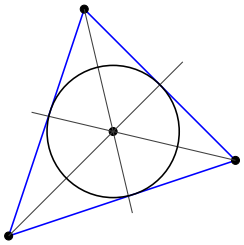
9.1.3 Centre du cercle circonscrit \tkzDefTriangleCenter[circum]



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

9.1.4 \tkzDefTriangleCenter[in]

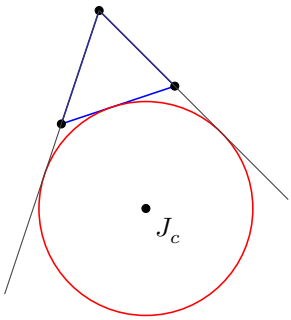
On obtient le centre du cercle inscrit du triangle. Le résultat est bien sûr dans **tkzPointResult**. On peut le récupérer avec **\tkzGetPoint**.



```
\begin{tikzpicture}
  \tkzDefPoints{0/1/A,3/2/B,1/4/C}
  \tkzDefTriangleCenter[in](A,B,C)\tkzGetPoint{I}
  \tkzDefPointBy[projection=onto A--C](I)
  \tkzGetPoint{Ib}
  \tkzDrawPolygon[color=blue](A,B,C)
  \tkzDrawPoints(A,B,C,I)
  \tkzDrawLines[add = 0 and 2/3](A,I B,I C,I)
  \tkzDrawCircle(I,Ib)
\end{tikzpicture}
```

9.1.5 \tkzDefTriangleCenter[ex]

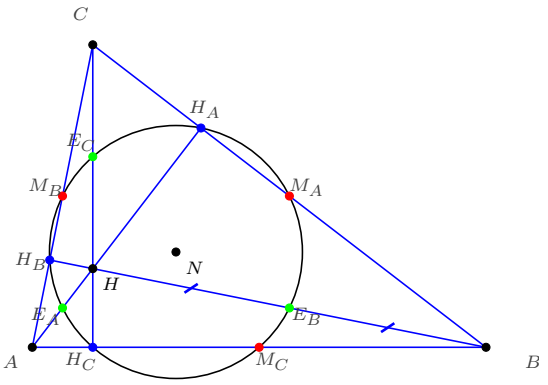
On obtient le centre d'un cercle inscrit du triangle. Le résultat est bien sûr dans **tkzPointResult**. On peut le récupérer avec **\tkzGetPoint**.




```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoints{0/1/A,3/2/B,1/4/C}
  \tkzDefCircle[ex](B,C,A)
  \tkzGetFirstPoint{J_c}
  \tkzGetSecondPoint{Tc}
  \tkzDrawPolygon[color=blue](A,B,C)
  \tkzDrawPoints(A,B,C,J_c)
  \tkzDrawCircle[red](J_c,Tc)
  \tkzDrawLines[add=1.5 and 0](A,C B,C)
  \tkzLabelPoints(J_c)
\end{tikzpicture}
```

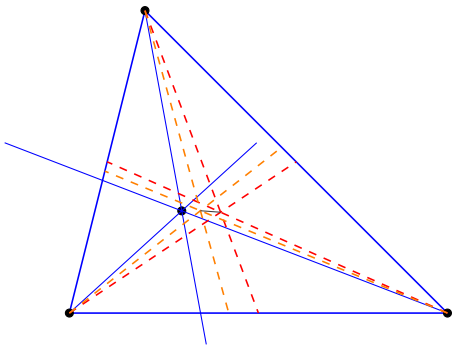
9.1.6 Utilisation de \tkzDefTriangleCenter[euler]

Cette macro permet d'obtenir le centre du cercle des neufs points ou cercle d'euler ou encore de Feuerbach.




```
\begin{tikzpicture}[scale=1]
 \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[color=blue](A,B,C)
 \tkzDrawCircle(N,E_A)
 \tkzDrawSegments[blue](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)
\end{tikzpicture}
```

9.1.7 Utilisation de \tkzDefTriangleCenter[symmedian]



```
\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[color=blue](A,B,C)
  \tkzDrawPoints(A,B,C,K)
  \tkzDrawLines[add = 0 and 2/3,blue](A,K B,K C,K)
  \tkzDrawSegments[red,dashed](A,Ma B,Mb C,Mc)
  \tkzDrawSegments[orange,dashed](A,Ia B,Ib C,Ic)
  \tkzDrawLine(G,I)
\end{tikzpicture}
```

10 Tracer un point

10.0.1 Tracer des points \tkzDrawPoint

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

arguments	défaut	définition
name of point	no default	Un seul nom de point est accepté

L'argument est obligatoire. Le disque prend la couleur du cercle mais 50% plus clair. Il est possible de tout modifier. Le point est un node et donc il est invariant si le dessin est modifié par une mise à l'échelle.

options	défaut	définition
shape	circle	Possible **cross** ou **cross out**
size	6	6× **\pgflinewidth**
color	black	la couleur par défaut peut être changée

On peut créer d'autres formes comme **cross**

10.0.2 Exemple de tracés de points

Il faut remarquer que **scale** ne touche pas à la forme des points. Ce qui est normal. La plupart du temps, on se contente d'une seule forme de points que l'on pourra définir dès le début, soit avec une macro, soit en modifiant un fichier de configuration.

```
\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[color=green](O)
\end{tikzpicture}
```

Il est possible de tracer plusieurs points en une seule fois mais cette macro est un peu plus lente que la précédente. De plus on doit se contenter des mêmes options pour tous les points.

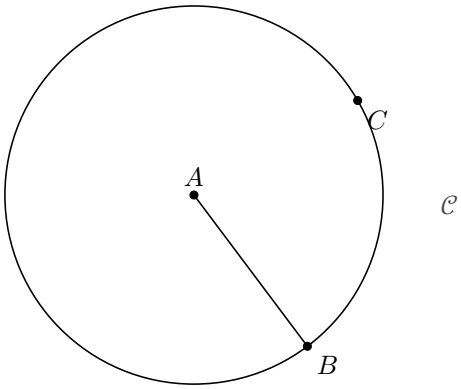
\tkzDrawPoints[⟨local options⟩](⟨liste⟩)

arguments	défaut	définition
liste de points	no default	exemple **\tkzDrawPoints(A,B,C)**

☞ Attention au « s » final, un oubli entraîne des erreurs en cascade si vous tentez de tracer des points multiples. Les options sont les mêmes que pour la macro précédente.

10.0.3 Exemple avec \tkzDefPoint et \tkzDrawPoints

```
\begin{tikzpicture}[scale=.5]
 \tkzDefPoint(1,3){A}
 \tkzDefPoint(4,1){B}
 \tkzDefPoint(0,0){O}
 \tkzDrawPoints[size=4,color=red](A,B,C)
\end{tikzpicture}
```




```
\begin{tikzpicture}[scale=.5]
 \tkzDefPoint(2,3){A}  \tkzDefPoint(5,-1){B}
 \tkzDefPoint[label=below:$\mathcal{C}$,
              shift={(2,3)}](-30:5.5){E}
 \begin{scope}[shift=(A)]
    \tkzDefPoint(30:5){C}
 \end{scope}
 \tkzCalcLength[cm](A,B)\tkzGetLength{rAB}
 \tkzDrawCircle[R](A,\rAB cm)
 \tkzDrawSegment(A,B)
 \tkzDrawPoints(A,B,C)
 \tkzLabelPoints(B,C)
 \tkzLabelPoints[above](A)
\end{tikzpicture}
```

11 Définition de points par transformation; \tkzDefPointBy

Ces transformations sont :

1. la translation ;
2. l'homothetie ;
3. la réflexion ou symétrie orthogonale ;
4. la symétrie centrale ;
5. la projection orthogonale ;
6. la rotation (degrés ou radians) ;
7. l'inversion par rapport à un cercle

Le choix des transformations se fait par l'intermédiaire des options. Il y a deux macros l'une pour la transformation d'un unique point **\tkzDefPointBy** et l'autre pour la transformation d'une liste de points **\tkzDefPointsBy**. Par défaut l'image de A est A'. Par exemple, on écrira :

```
\tkzDefPointBy[translation= from A to A'](B) le résultat est dans tkzPointResult}
```

\tkzDefPointBy[⟨local options⟩](⟨pt⟩)

L'argument est un simple point existant et son image est stockée dans **tkzPointResult**. Si vous voulez conserver ce point alors la macro **\tkzGetPoint{M}** permet d'attribuer le nom **M** au point.

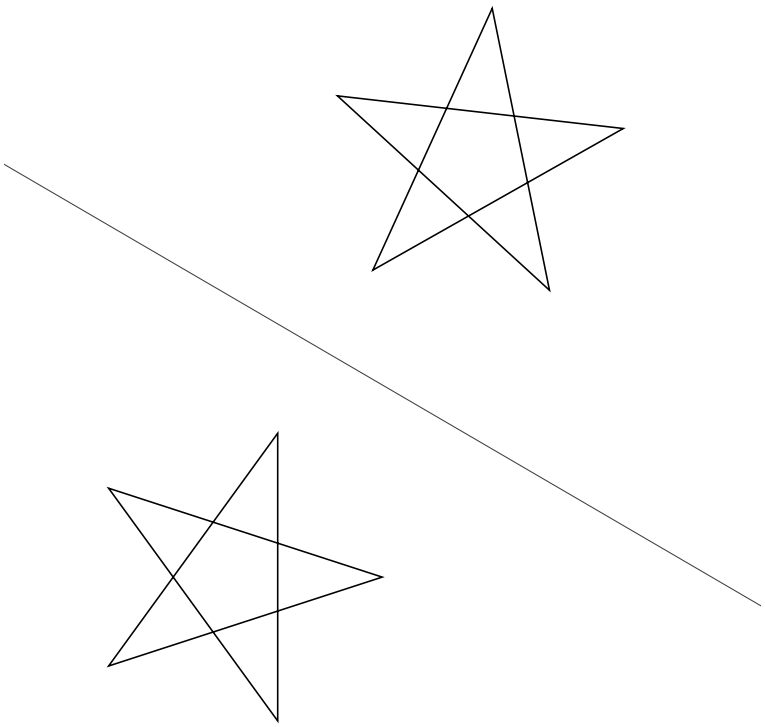
arguments	définition	exemples
pt	nom d'un point existant	(A)

options		exemples
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=center O angle pi/3
inversion	= center #1 through #2	[inversion =center O through A](E)

L'image est seulement définie et non tracée.

11.1 La réflexion ou symétrie orthogonale

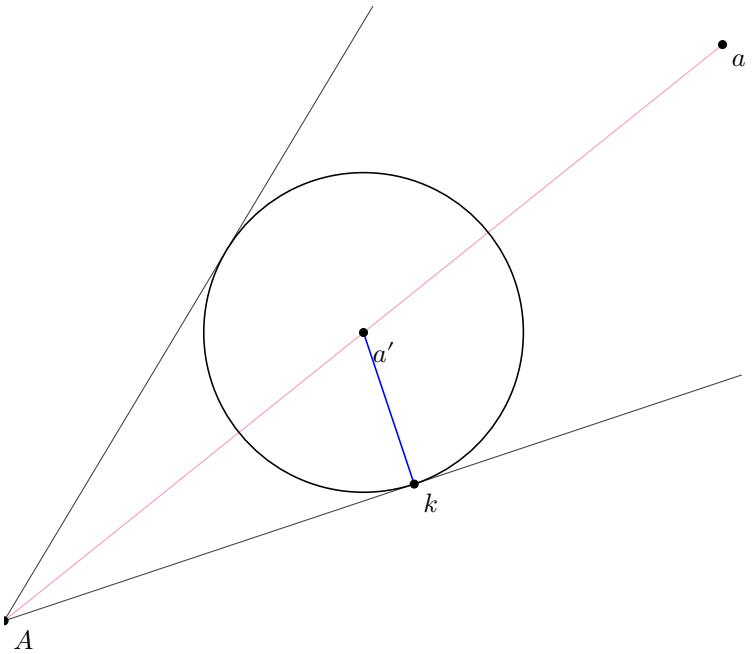
11.1.1 Exemple de réflexion



```
\begin{tikzpicture}[scale=1]
 \tkzInit[ymin=-4,ymax=6,xmin=-7,xmax=3]
 \tkzClip
 \tkzDefPoints{1.5/-1.5/C,-4.5/2/D}
 \tkzDefPoint(-4,-2){O}
 \tkzDefPoint(-2,-2){A}
 \foreach \i in {0,1,...,4}{%
 \pgfmathparse{0+\i * 72}
 \tkzDefPointBy[rotation=center O angle \pgfmathresult](A) \tkzGetPoint{A\i}
 \tkzDefPointBy[reflection = over C--D](A\i) \tkzGetPoint{A\i'}}
 \tkzDrawPolygon(A0, A2, A4, A1, A3)
 \tkzDrawPolygon(A0', A2', A4', A1', A3')
 \tkzDrawLine[add= .5 and .5](C,D)
\end{tikzpicture}
```

11.2 L'homothétie

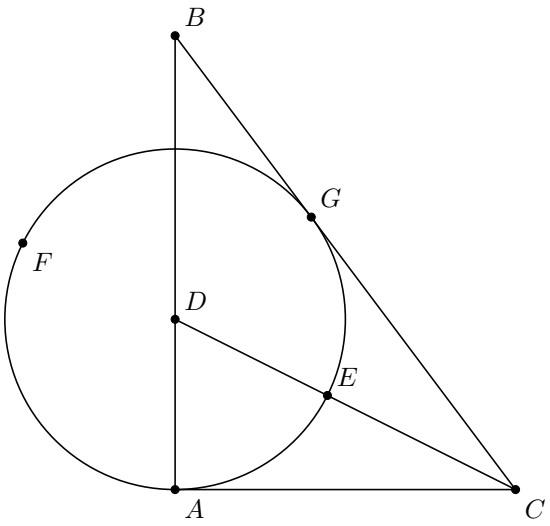
11.2.1 Exemple d'homothétie et de projection




```
\begin{tikzpicture}[scale=1.25]
  \tkzInit   \tkzClip
  \tkzDefPoint(0,1){A}   \tkzDefPoint(6,3){B}   \tkzDefPoint(3,6){C}
  \tkzDrawLines[add= 0 and .3](A,B A,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}
  \tkzDrawSegment[blue](a',k)
  \tkzDrawPoints(a,a',k,A)
  \tkzDrawCircle(a',k)
  \tkzLabelPoints(a,a',k,A)
\end{tikzpicture}
```

11.3 La projection

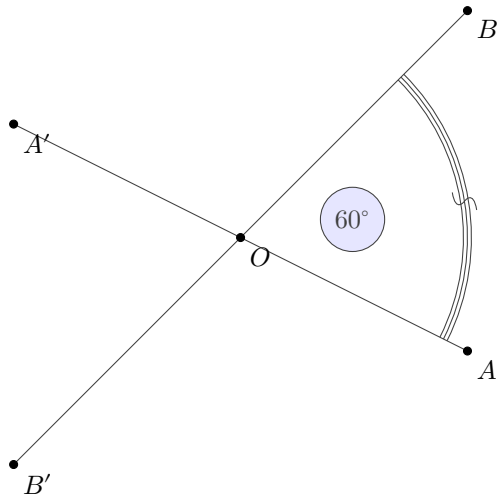
11.3.1 Exemple de projection




```
\begin{tikzpicture}[scale=1.5]
 \tkzInit[xmin=-3,xmax=5,ymax=4] \tkzClip[space=.5]
 \tkzDefPoint(0,0){A}
 \tkzDefPoint(0,4){B}
 \tkzDrawTriangle[pythagore](B,A) \tkzGetPoint{C}
 \tkzDefLine[bisector](B,C,A) \tkzGetPoint{c}
 \tkzInterLL(C,c)(A,B)        \tkzGetPoint{D}
 \tkzDrawSegment(C,D)
 \tkzDrawCircle(D,A)
 \tkzDefPointBy[projection=onto B--C](D) \tkzGetPoint{G}
 \tkzInterLC(C,D)(D,A) \tkzGetPoints{E}{F}
 \tkzDrawPoints(A,C,F) \tkzLabelPoints(A,C,F)
 \tkzDrawPoints(B,D,E,G)
 \tkzLabelPoints[above right](B,D,E,G)
 \end{tikzpicture}
```

11.4 La symétrie

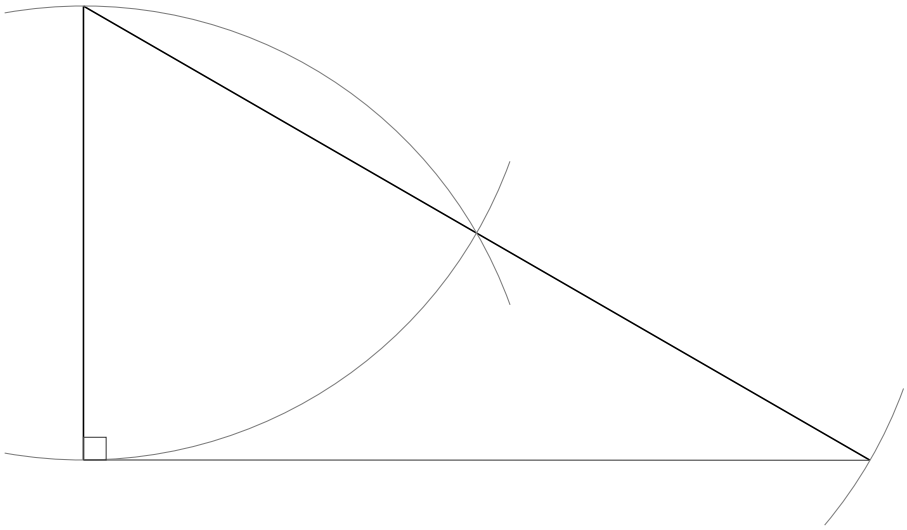
11.4.1 Exemple de symétrie




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

11.5 La rotation

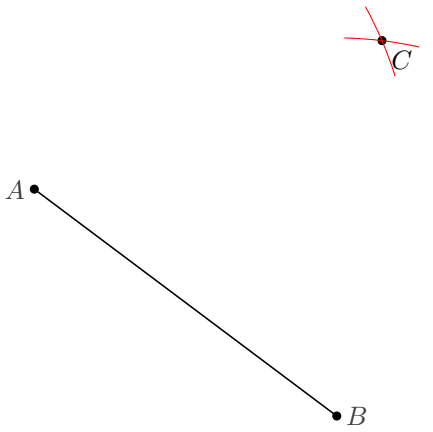
11.5.1 Exemple de rotation



```
\begin{tikzpicture}[scale=1.2,rotate=-90]
\tkzInit
\tkzDefPoint(0,0){A} \tkzDefPoint(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[delta=10](A,B)(C)
\tkzDrawArc[delta=10](B,C)(A)
\tkzDrawArc[delta=10](C,D)(D)
\tkzMarkRightAngle(D,B,A)
\end{tikzpicture}
```

11.6 La rotation en radian

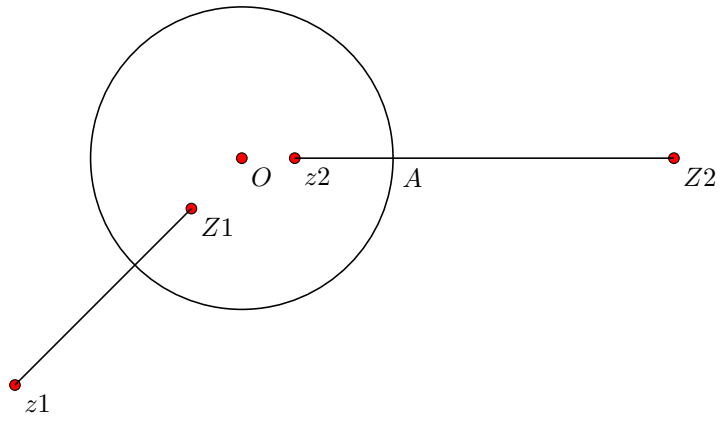
11.6.1 Exemple de rotation en radian




```
\begin{tikzpicture}
  \tkzDefPoint["$A$" left](1,5){A}
  \tkzDefPoint["$B$" right](5,2){B}
  \tkzDefPointBy[rotation in rad= center A angle pi/3](B)
  \tkzGetPoint{C}

  \tkzDrawSegment(A,B)
  \tkzDrawPoints(A,B,C)
  \tkzCompass[color=red](A,C)
  \tkzCompass[color=red](B,C)

  \tkzLabelPoints(C)
\end{tikzpicture}
```

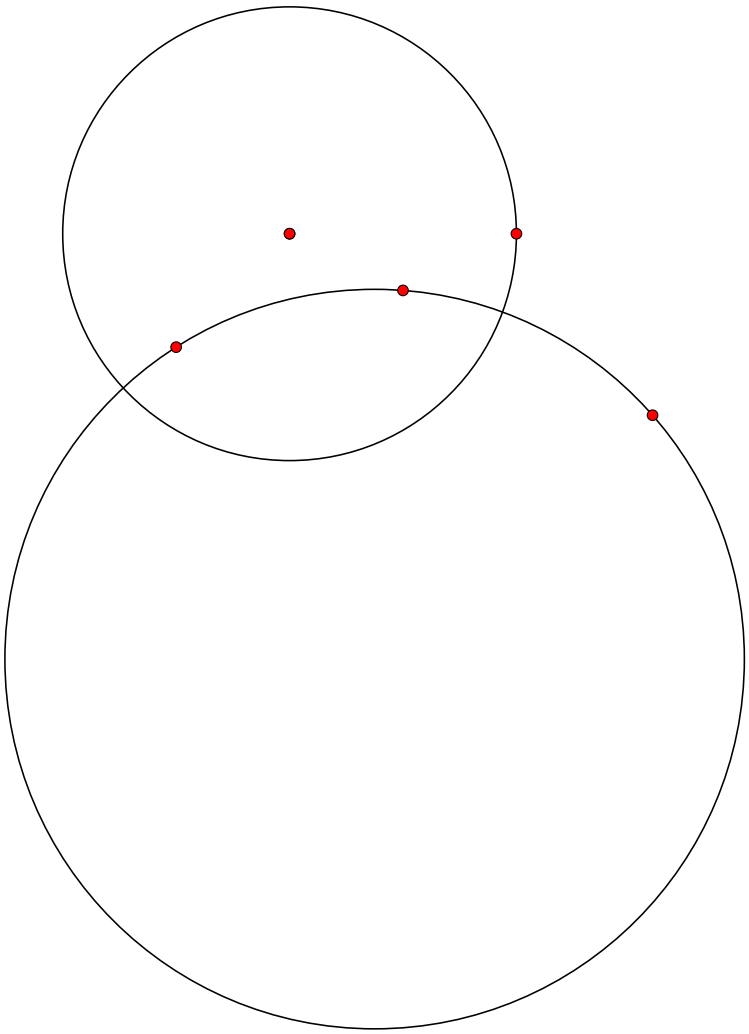
11.7 L'inversion par rapport à un cercle

11.7.1 Inversion de points




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

11.7.2 Inversion de point : cercles orthogonaux



```
\begin{tikzpicture}[scale=3]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(1,0){A}
  \tkzDrawCircle(O,A)
  \tkzDefPoint(0.5,-0.25){z1}
  \tkzDefPoint(-0.5,-0.5){z2}
  \tkzDefPointBy[inversion = center O through A](z1)
  \tkzGetPoint{Z1}
  \tkzCircumCenter(z1,z2,Z1)\tkzGetPoint{c}
  \tkzDrawCircle(c,Z1)
  \tkzDrawPoints[color=black,fill=red,size=4](O,z1,z2,Z1,O,A)
\end{tikzpicture}
```

12 Transformation de multiples points; \tkzDefPointsBy

Variante de la précédente macro pour la définition de multiples images. Il faut donner en argument, les noms des images ou bien encore indiquer que le nom des images est formé à partir du nom des antécédents en laissant l'argument vide.

```
\tkzDefPointsBy[translation= from A to A'](B,C){} les images sont B' et C'
\tkzDefPointsBy[translation= from A to A'](B,C){D,E} les images sont D et E
\tkzDefPointsBy[translation= from A to A'](B) l'image est B'
```

\tkzDefPointsBy[⟨local options⟩](⟨liste de pts⟩){⟨liste de pts⟩}

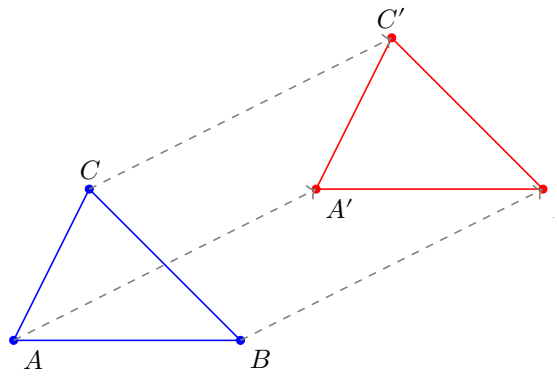
arguments	exemples	
(⟨liste de pts⟩){⟨liste de pts⟩}	(A,B){E,F}	E est l'image de A et F celle de B.

Si la liste des images est vide alors le nom de l'image est le nom de l'antécédent auquel on ajoute « ' »

options	exemples
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	par exemple angle pi/3

Les points sont seulement définis et non tracés.

12.1 Exemple de translation




```
\begin{tikzpicture}
 \tkzDefPoint(0,0){A}  \tkzDefPoint(4,2){A'}
 \tkzDefPoint(3,0){B}  \tkzDefPoint(1,2){C}
 \tkzDefPointsBy[translation= from A to A'](B,C){}
 \tkzDrawPolygon[color=blue](A,B,C)
 \tkzDrawPolygon[color=red](A',B',C')
 \tkzDrawPoints[color=blue](A,B,C)
 \tkzDrawPoints[color=red](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}
```

13 Définition de points à l'aide d'un vecteur

13.1 \tkzDefPointWith

Il y a plusieurs possibilités pour créer des points qui répondent à certaines conditions vectorielles. Cela peut se faire avec **\tkzDefPointWith**. Le principe général est le suivant, deux points sont passés en argument, autrement dit un vecteur. Les différentes options permettent d'obtenir un nouveau point formant avec le premier point (sauf exception) un vecteur colinéaire ou bien orthogonal au premier vecteur. Ensuite la longueur est soit proportionnelle à celle du premier, ou bien proportionnelle à l'unité. Dans la mesure ou ce point n'est utilisé que temporairement, il n'est pas obligé de le nommer immédiatement. Le résultat est dans **\tkzPointResult**. La macro **\tkzGetPoint** permet de récupérer le point et de le nommer différemment.

Des options permettent de définir la distance entre le point donné et le point obtenu. Dans le cas général cette distance est celle entre les 2 points donnés en arguments si l'option est du type "normed" alors la distance entre le point donné et le point obtenu.est de 1 cm. Ensuite l'option K permet d'obtenir des multiples.

\tkzDefPointWith(⟨pt1,pt2⟩)

Il s'agit en fait de la définition d'un point répondant à des conditions vectorielles.

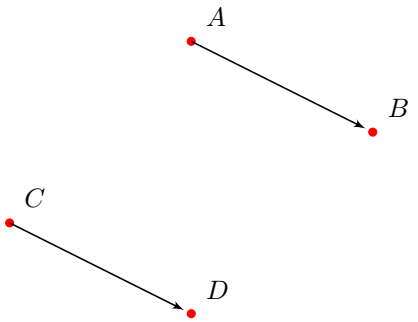
arguments	définition	explication
(pt1,pt2)	couple de points	le résultat est un point dans **\tkzPointResult**

Dans ce qui suit, on suppose que le point est récupéré par **\tkzGetPoint{C}**

options	exemple	explication
orthogonal	[orthogonal](A,B)	$AC = AB$ et $\overrightarrow{AC} \perp \overrightarrow{AB}$
orthogonal normed	[orthogonal normed](A,B)	$AC = 1$ et $\overrightarrow{AC} \perp \overrightarrow{AB}$
linear	[linear](A,B)	$\overrightarrow{AC} = K \times \overrightarrow{AB}$
linear normed	[linear normed](A,B)	$AC = K$ et $\overrightarrow{AC} = k \times \overrightarrow{AB}$
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	$\overrightarrow{AC} = 2 \times \overrightarrow{AB}$

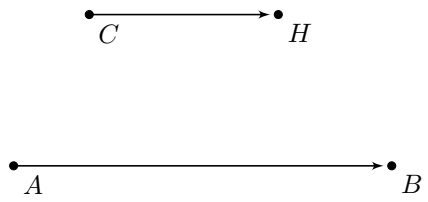
13.1.1 \tkzDefPointWith et colinear at

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



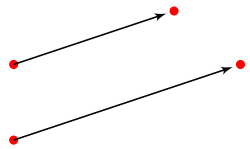

```
\begin{tikzpicture}[scale=1.2,
   vect/.style={->,shorten >=3pt,>=latex'}]
  \tkzDefPoint(2,3){A}   \tkzDefPoint(4,2){B}
  \tkzDefPoint(0,1){C}
  \tkzDefPointWith[colinear=at C](A,B)
  \tkzGetPoint{D}
  \tkzDrawPoints[color=red](A,B,C,D)
  \tkzLabelPoints[above right=3pt](A,B,C,D)
  \tkzDrawSegments[vect](A,B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

13.1.2 colinear at




```
\begin{tikzpicture}[vect/.style={->,
              shorten >=3pt,>=latex'}]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(5,0){B}
  \tkzDefPoint(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}
```

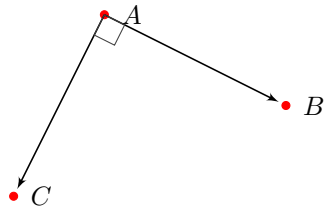
13.1.3 colinear $K = \frac{\sqrt{2}}{2}$



```
\begin{tikzpicture}[vect/.style={->,
          shorten >=3pt,>=latex'}]
 \tkzDefPoint(1,1){A}
 \tkzDefPoint(4,2){B}
 \tkzDefPoint(2,2){CU}
 \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}
```

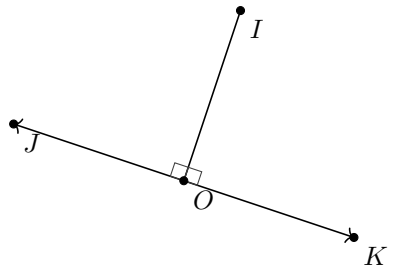
13.1.4 \tkzDefPointWith et orthogonal

$K = -1$ afin que $(\overrightarrow{AC}, \overrightarrow{AB})$ détermine un angle positif. AB=AC puisque $\mathtt{K} = 1$



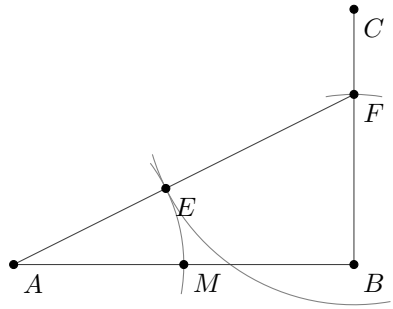

```
\begin{tikzpicture}[scale=1.2,
  vect/.style={->,shorten >=3pt,>=latex'}]
   \tkzDefPoint(2,3){A}
   \tkzDefPoint(4,2){B}
   \tkzDefPointWith[orthogonal,K=-1](A,B)
     \tkzGetPoint{C}
   \tkzDrawPoints[color=red](A,B,C)
   \tkzLabelPoints[right=3pt](A,B,C)
   \tkzDrawSegments[vect](A,B A,C)
   \tkzMarkRightAngle(B,A,C)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

13.1.5 orthogonal simple



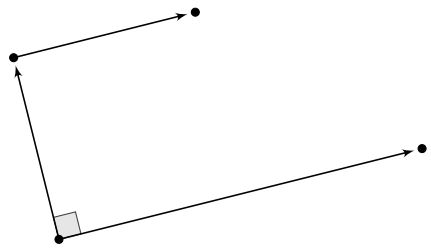

```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoint(1,2){O}
  \tkzDefPoint(2,5){I}
  \tkzDefPointWith[orthogonal](O,I)
  \tkzGetPoint{J}
  \tkzDefPointWith[orthogonal,K=-1](O,I)
  \tkzGetPoint{K}
  \tkzDrawSegment(O,I)
  \tkzDrawSegments[->](O,J O,K)
  \tkzMarkRightAngles(I,O,J I,O,K)
  \tkzDrawPoints(O,I,J,K)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

13.1.6 orthogonal avancé




```
\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,E)
\end{tikzpicture}
```

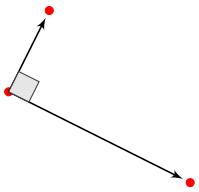
13.1.7 segment colinear et orthogonal



```
\begin{tikzpicture}[scale=1.2,
  vect/.style={->,shorten >=3pt,>=latex'}]
   \tkzDefPoint(2,1){A}
   \tkzDefPoint(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}
```

13.1.8 \tkzDefPointWith orthogonal normed, K=1

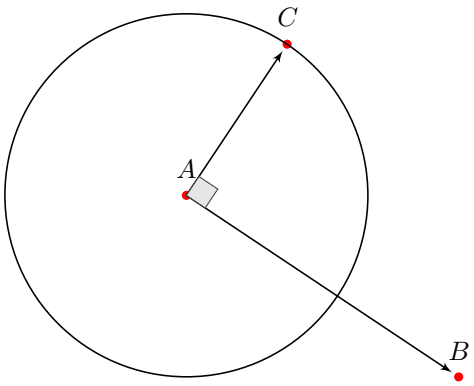
AC=1



```
\begin{tikzpicture}[scale=1.2,
  vect/.style={->,shorten >=3pt,>=latex'}]
  \tkzDefPoint(2,3){A}   \tkzDefPoint(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}
```

13.1.9 \tkzDefPointWith et orthogonal normed K=2

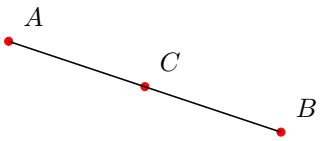
$K = 2$ donc AC=2.




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

13.1.10 \tkzDefPointWith linear

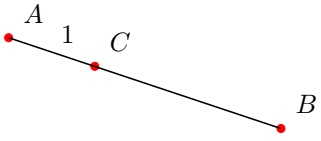
Ici $K = 0.5$ Cela revient à appliquer une homothétie ou bien encore une multiplication d'un vecteur par un réel. C est ici le milieu de $[AB]$.




```
\begin{tikzpicture}[scale=1.2]
   \tkzDefPoint(1,3){A}   \tkzDefPoint(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}
```

13.1.11 \tkzDefPointWith linear normed

Dans l'exemple suivant AC=1 et C appartient à (AB).




```
\begin{tikzpicture}[scale=1.2]
 \tkzDefPoint(1,3){A}   \tkzDefPoint(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}
```

13.2 \tkzGetVectxy

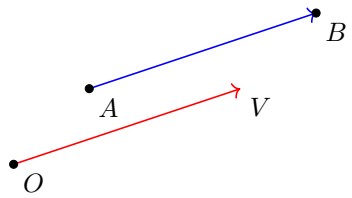
Récupération des coordonnées d'un vecteur

\tkzGetVectxy(⟨A, B⟩){⟨text⟩}

Permet d'obtenir les coordonnées d'un vecteur

arguments	exemple	explication
(point){name of macro}	\tkzGetVectxy(A,B){V}	\Vx,\Vy : coordonnées de $\overrightarrow{AB}$

13.2.1 Transfert de coordonnées avec \tkzGetVectxy




```
\begin{tikzpicture}
 \tkzDefPoint(0,0){O}
 \tkzDefPoint(1,1){A}
 \tkzDefPoint(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}
```

14 Définition aléatoire de points

Il y a pour le moment quatre possibilités :

1. point dans un rectangle,
2. sur un segment,
3. sur une droite,
4. sur un cercle.

14.1 Obtention de points aléatoirement

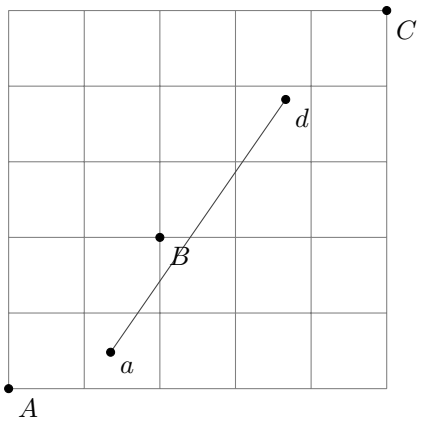
Ceci est la nouvelle version qui remplace \tkzGetRandPointOn

\tkzDefRandPointOn[⟨local options⟩]

Le résultat est un point avec une position aléatoire que l'on peut nommer avec la macro **\tkzGetPoint**. Il est possible d'utiliser **tkzPointResult** s'il n'est pas nécessaire de conserver les résultats.

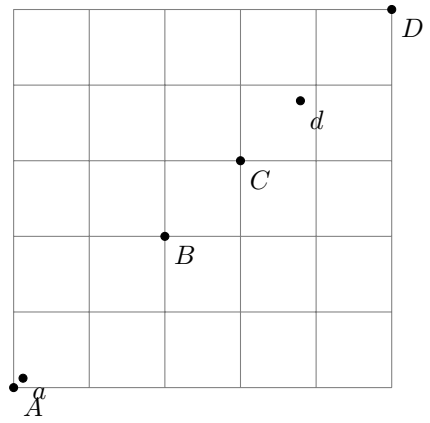
options	défaut	définition
rectangle=pt1 and pt2		[rectangle=A and B]
segment= pt1--pt2		[segment=A--B]
line=pt1--pt2		[line=A--B]
circle =center pt1 radius dim		[circle = center A radius 2cm]
circle through=center pt1 through pt2		[circle through= center A through B]
disk through=center pt1 through pt2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

14.2 Point aléatoire dans un rectangle



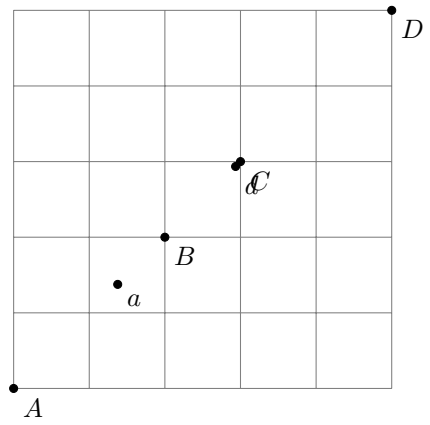

```
\begin{tikzpicture}
  \tkzInit[xmax=5,ymax=5]\tkzGrid
  \tkzDefPoints{0/0/A,2/2/B,5/5/C}
  \tkzDefRandPointOn[rectangle = A and B]
  \tkzGetPoint{a}
  \tkzDefRandPointOn[rectangle = B and C]
  \tkzGetPoint{d}
  \tkzDrawLine(a,d)
  \tkzDrawPoints(A,B,C,a,d)
  \tkzLabelPoints(A,B,C,a,d)
\end{tikzpicture}
```

14.3 Point aléatoire sur un segment



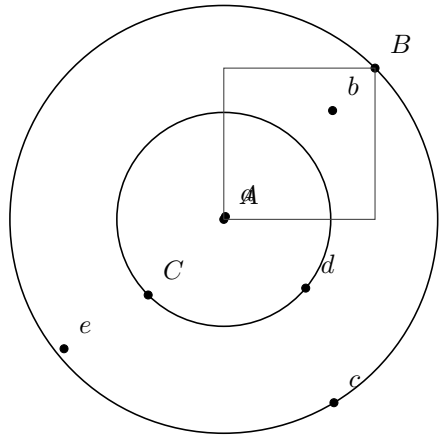

```
\begin{tikzpicture}
  \tkzInit[xmax=5,ymax=5] \tkzGrid
  \tkzDefPoints{0/0/A,2/2/B,3/3/C,5/5/D}
  \tkzDefRandPointOn[segment = A--B]\tkzGetPoint{a}
  \tkzDefRandPointOn[segment = C--D]\tkzGetPoint{d}
  \tkzDrawPoints(A,B,C,D,a,d)
  \tkzLabelPoints(A,B,C,D,a,d)
\end{tikzpicture}
```

14.4 Point aléatoire sur une droite




```
\begin{tikzpicture}
  \tkzInit[xmax=5,ymax=5] \tkzGrid
  \tkzDefPoints{0/0/A,2/2/B,3/3/C,5/5/D}
  \tkzDefRandPointOn[line = A--B]\tkzGetPoint{a}
  \tkzDefRandPointOn[line = C--D]\tkzGetPoint{d}
  \tkzDrawPoints(A,B,C,D,a,d)
  \tkzLabelPoints(A,B,C,D,a,d)
\end{tikzpicture}
```

14.4.1 Exemple de points aléatoires




```
\begin{tikzpicture}
 \tkzDefPoints{0/0/A,2/2/B,-1/-1/C}
 \tkzDefCircle[through=](A,C)
 \tkzGetLength{rAC}
 \tkzDrawCircle(A,C)
 \tkzDrawCircle(A,B)
 \tkzDefRandPointOn[rectangle=A and B]
 \tkzGetPoint{a}
 \tkzDefRandPointOn[segment=A--B]
 \tkzGetPoint{b}
 \tkzDefRandPointOn[circle=center A radius \rAC pt]
    \tkzGetPoint{d}
 \tkzDefRandPointOn[circle through= center A through B]
     \tkzGetPoint{c}
 \tkzDefRandPointOn[disk through=center A through B]
     \tkzGetPoint{e}
 \tkzLabelPoints[above right=3pt](A,B,C,a,b,...,e)
 \tkzDrawPoints[](A,B,C,a,b,...,e)
 \tkzDrawRectangle(A,B)
\end{tikzpicture}
```

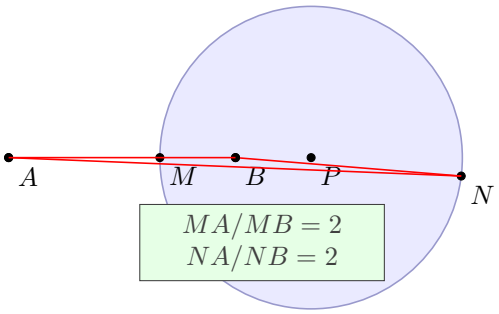
14.5 Obtention de points aléatoirement \tkzGetRandPointOn

Ceci est l'ancienne version. Le point obtenu doit être nommé.

\tkzDefRandPointOn[⟨local options⟩]{⟨name⟩}

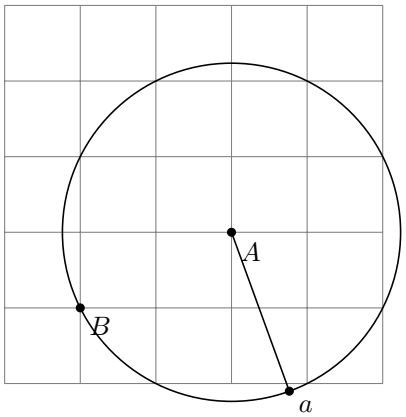
Le résultat est un point avec une position aléatoire que l'on peut nommer avec la macro **\tkzGetPoint**. Il est possible d'utiliser **tkzPointResult** s'il n'est pas nécessaire de conserver les résultats. Les options sont les mêmes que la précédente.

14.5.1 Exemple random et cercle d'Apollonius




```
\begin{tikzpicture}[scale=1]
 \tkzDefPoints{0/0/A,3/0/B}
 \def\coeffK{2}
 \tkzApolloniusCenter[K=\coeffK](A,B)
 \tkzGetPoint{P}
 \tkzDefApolloniusPoint[K=\coeffK](A,B)
 \tkzGetPoint{M}
 \tkzDefApolloniusRadius[K=\coeffK](A,B)
 \tkzDrawCircle[R,color = blue!50!black,
     fill=blue!20,
     opacity=.4](tkzPointResult,\tkzLengthResult pt)
 \tkzDefRandPointOn[circle through= center P through M]
 \tkzGetPoint{N}
 \tkzDrawPoints(A,B,P,M,N)
 \tkzLabelPoints(A,B,P,M,N)
 \tkzDrawSegments[red](N,A N,B)
 \tkzDrawPoints(A,B)
 \tkzDrawSegments[red](A,B)
 \tkzLabelCircle[R,draw,fill=green!10,%
     text width=3cm,%
     text centered](P,\tkzLengthResult pt-20pt)(-
120)%
  { $MA/MB=\coeffK$\\$NA/NB=\coeffK$}
\end{tikzpicture}
```

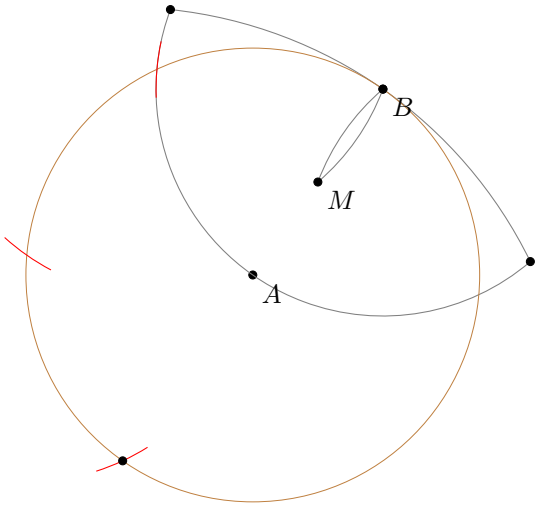
14.6 Point aléatoire sur un cercle




```
\begin{tikzpicture}
  \tkzInit[xmax=5,ymax=5]  \tkzGrid
  \tkzDefPoints{3/2/A,1/1/B}
  \tkzCalcLength[cm](A,B) \tkzGetLength{rAB}
  \tkzDrawCircle[R](A,\rAB cm)
  \tkzDefRandPointOn[circle = center A radius
   \rAB cm]\tkzGetPoint{a}
  \tkzDrawSegment(A,a)
  \tkzDrawPoints(A,B,a)
  \tkzLabelPoints(A,B,a)
\end{tikzpicture}
```

14.7 Milieu d'un segment au compas

Pour terminer cette section, voici un exemple plus complexe. Il s'agit de déterminer le milieu d'un segment, uniquement avec un compas.




```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoint(0,0){A}
  \tkzDefRandPointOn[circle= center A radius 4cm]
  \tkzGetPoint{B}
  \tkzDrawPoints(A,B)
  \tkzDefPointBy[rotation= center A angle 180](B)
  \tkzGetPoint{C}
  \tkzInterCC[R](A,4 cm)(B,4 cm)
  \tkzGetPoints{I}{I'}
  \tkzInterCC[R](A,4 cm)(I,4 cm)
  \tkzGetPoints{J}{B}
  \tkzInterCC(B,A)(C,B)
  \tkzGetPoints{D}{E}
  \tkzInterCC(D,B)(E,B)
  \tkzGetPoints{M}{M'}
  \tikzset{arc/.style={color=brown,style=dashed,delta=10}}
  \tkzDrawArc[arc](C,D)(E)
  \tkzDrawArc[arc](B,E)(D)
  \tkzDrawCircle[color=brown,line width=.2pt](A,B)
  \tkzDrawArc[arc](D,B)(M)
  \tkzDrawArc[arc](E,M)(B)
  \tkzCompasss[color=red,style=solid](B,I I,J J,C)
  \tkzDrawPoints(B,C,D,E,M)
  \tkzLabelPoints(A,B,M)
 \end{tikzpicture}
```

15 Les droites

Il est bien sûr essentiel de tracer des droites, mais avant il faut pouvoir définir certaines droites particulières comme des médiatrices, des bissectrices, des parallèles ou encore des perpendiculaires. Le principe consiste à déterminer deux points de la droite.

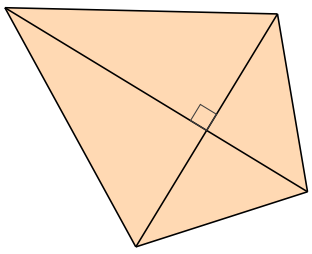
15.1 Définition de droites

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

L' argument est une liste de deux ou trois points. Suivant les cas, la macro définit un ou deux points nécessaires pour obtenir la droite cherchée. Il faut utiliser soit la macro **\tkzGetPoint**, soit la macro **\tkzGetPoints**.

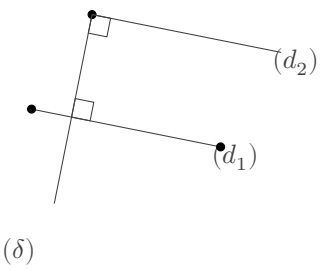
options	défaut	définition
mediator		médiatrice. Deux points sont définis
perpendicular=through…		perpendiculaire à une droite passant par un point
orthogonal=through…		voir ci-dessus
parallel=through…		parallèle à une droite passant par un point
bisector		bissectrice d'un angle défini par trois points
bisector out		bissectrice extérieure
tangent=at…		tangente à un cercle en un point donné
tangent=from…		tangente à un cercle(O,A) passant par un point donné
tangent=from with R…		tangente à un cercle(O,r) passant par un point donné
K	1	Coefficient pour la droite perpendiculaire

15.1.1 Exemple avec mediator



```
\begin{tikzpicture}[rotate=25]
  \tkzInit
  \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=orange!30](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}
```

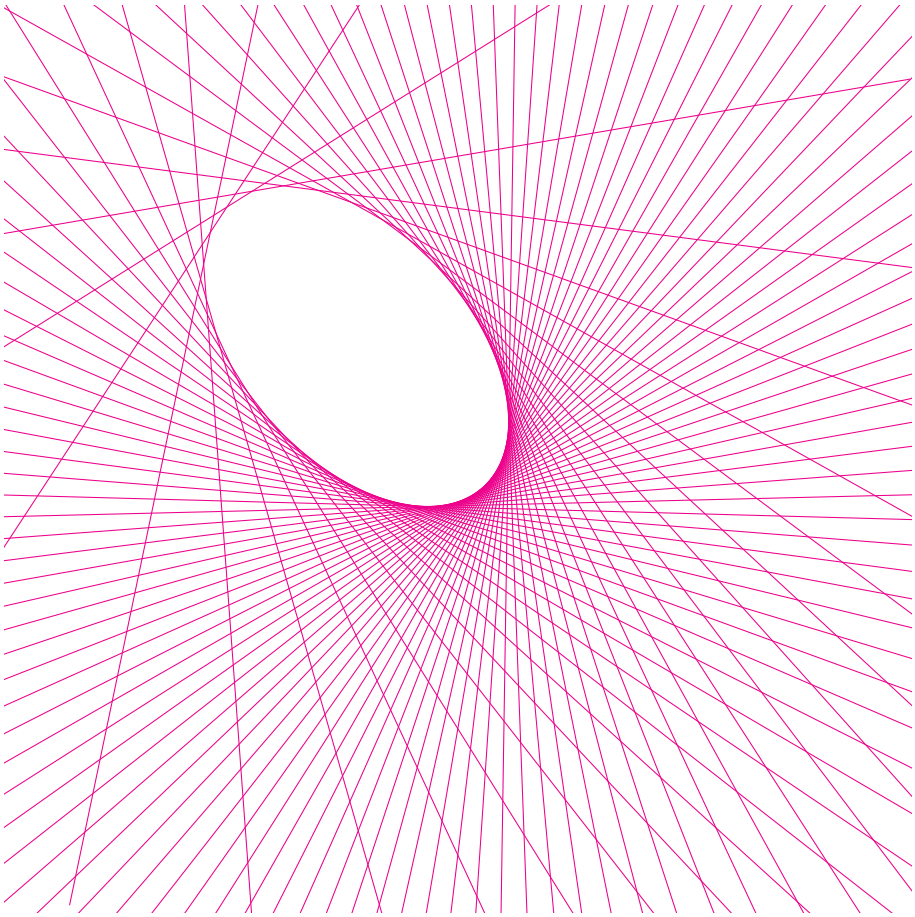
15.1.2 Exemple avec orthogonal et parallel




```
\begin{tikzpicture}
   \tkzDefPoints{-1.5/-0.25/A,1/-0.75/B,-0.7/1/C}
   \tkzDrawLine(A,B)
   \tkzLabelLine[pos=1.25,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,left](C,c'){$(d_2)$}
   \tkzMarkRightAngle(I,C,c')
\end{tikzpicture}
```

15.1.3 Une enveloppe

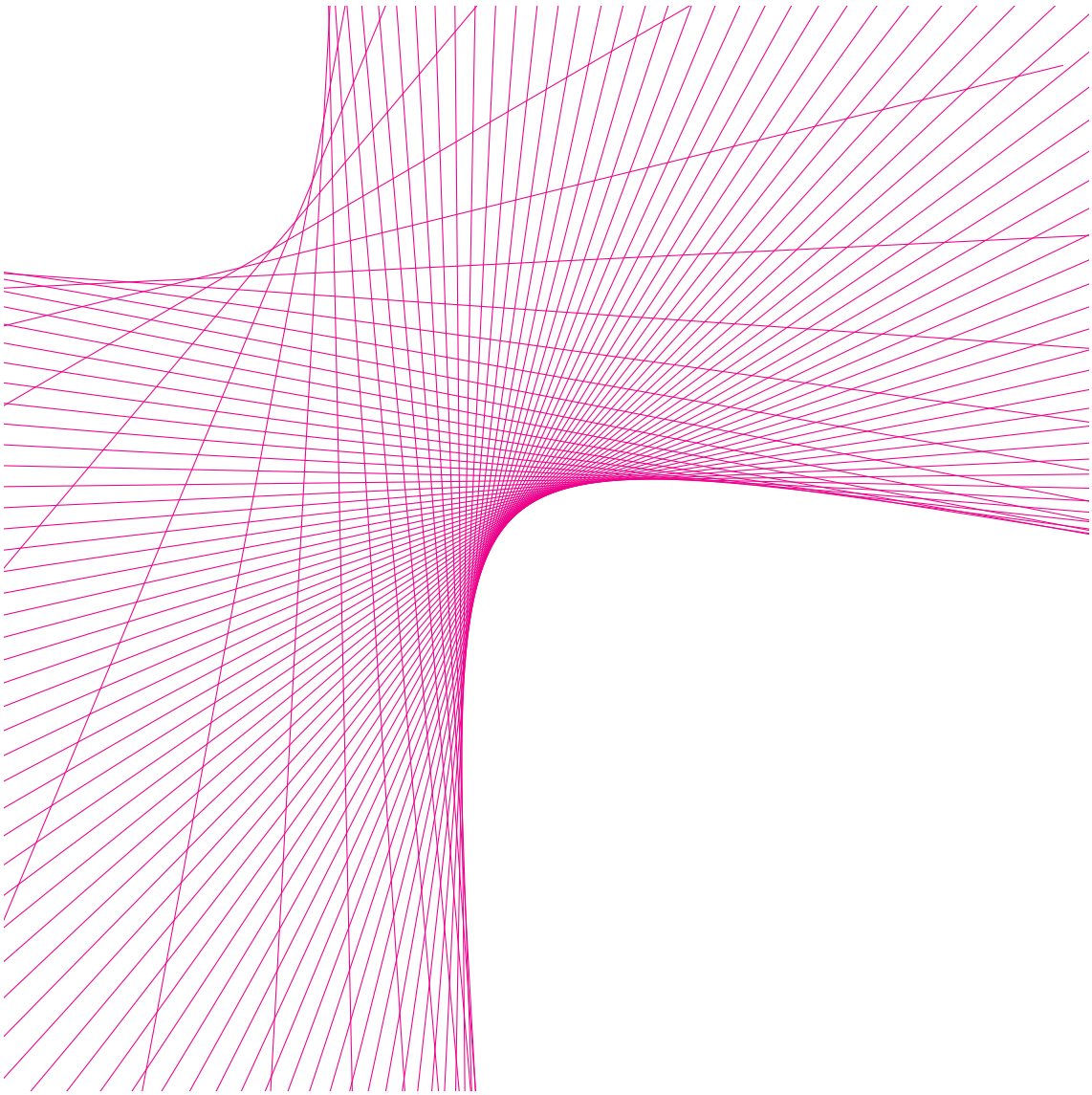
D'après une figure d'O. Reboux avec pst-eucl de D Rodriguez



```
\begin{tikzpicture}[scale=1]
  \tkzInit[xmin=-6,ymin=-6,xmax=6,ymax=6]
  \tkzClip
  \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)
    \tkzDrawLine[color=magenta,add= 4 and 4](tkzFirstPointResult,tkzSecondPointResult)}
\end{tikzpicture}
```

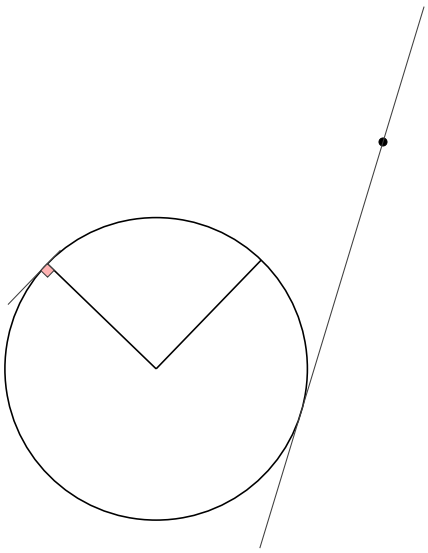
15.1.4 Une parabole

D'après une figure d'O. Reboux avec pst-eucl de D Rodriguez. Il n'est pas nécessaire de nommer les deux points qui définissent la médiatrice.



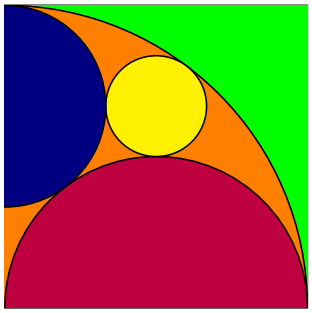
```
\begin{tikzpicture}[scale=1.25]
  \tkzInit[xmin=-6,ymin=-6,xmax=6,ymax=6]
  \tkzClip
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(132:5){A}
  \tkzDefPoint(4,0){B}
  \foreach \ang in {5,10,...,360}{%
    \tkzDefPoint(\ang:4){M}
    \tkzDefLine[mediator](A,M)
    \tkzDrawLine[color=magenta,
              add= 4 and 4](tkzFirstPointResult,tkzSecondPointResult)}
   \end{tikzpicture}
```

15.1.5 Tracer une tangente option from with R and at



```
\begin{tikzpicture}[scale=.5]
\tkzDefPoint(0,0){O}
\tkzDefPoint(6,6){E}
\tkzDefRandPointOn[circle=center O radius 4cm]
\tkzGetPoint{A}
\tkzDefRandPointOn[circle=center O radius 4cm]
\tkzGetPoint{B}
\tkzDrawSegments(O,A O,B)
\tkzDrawCircle(O,A)
\tkzDefTangent[from with R=E](O,4cm)
\tkzGetSecondPoint{k}
\tkzDefTangent[at=A](O)
\tkzGetPoint{h}
\tkzDrawPoints(E)
\tkzDrawLine[add = .5 and .5](A,h)
\tkzDrawLine[add = .5 and .5](E,k)
\tkzMarkRightAngle[fill=red!30](O,A,h)
\end{tikzpicture}
```

15.1.6 Tracer une tangente option from



```
\begin{tikzpicture}[scale=.5]
 \tkzDefPoint(0,0){B}
 \tkzDefPoint(0,8){A}
 \tkzDefSquare(A,B)
 \tkzGetPoints{C}{D}
 \tkzDrawSquare(A,B)
 \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)
 \tkzDefTangent[from=B](F,A)
 \tkzInterLL(F,tkzFirstPointResult)(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}
```

16 Tracer, nommer les droites

Les macros suivantes permettent simplement de tracer, de nommer des droites

16.1 Tracer une droite

Pour tracer une droite, il suffit de donner les deux points et d'utiliser l'option **add**. Cette option est due à Mark Wibrow

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

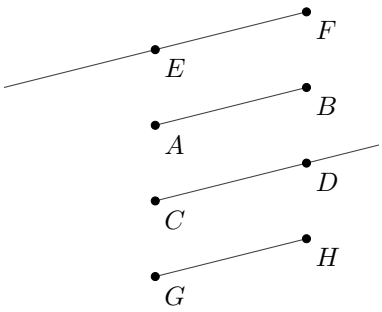
\tkzDrawLine[⟨local options⟩](⟨pt1,pt2⟩) ou (⟨pt1,pt2,pt3⟩)

Les arguments sont une liste de deux points.

options	défaut	définition
median	none	[median](A,B,C) médiane issue de B
altitude	none	[altitude](C,A,B) hauteur issue de A
bisector	none	[bisector](B,C,A) bissectrice issue de C
none	none	(A,B)
add= nb1 and nb2	.2 and .2	Permet de prolonger le segment

add permet de définir la longueur du trait passant par les points pt1 et pt2. Les deux nombres sont des pourcentages. Les styles de Ti*k*Z sont accessibles pour les tracés

16.1.1 Exemples de tracés de droite avec add



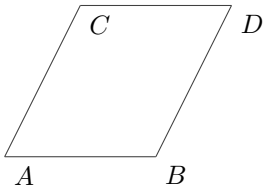

```
\begin{tikzpicture}
 \tkzInit[xmin=-2,xmax=3,ymin=-2.25,ymax=2.25]
 \tkzClip[space=.25]
 \tkzDefPoint(0,0){A} \tkzDefPoint(2,0.5){B}
 \tkzDefPoint(0,-1){C}\tkzDefPoint(2,-0.5){D}
 \tkzDefPoint(0,1){E} \tkzDefPoint(2,1.5){F}
 \tkzDefPoint(0,-2){G} \tkzDefPoint(2,-1.5){H}
 \tkzDrawLine(A,B)    \tkzDrawLine[add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

Il est possible de tracer plusieurs droites, mais avec les mêmes options.

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

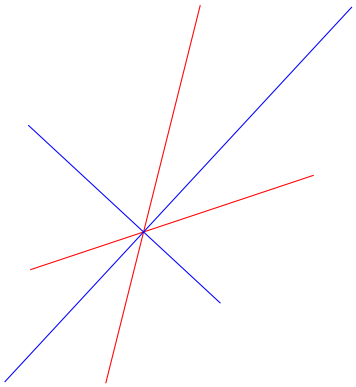
Les arguments sont une liste de couples de deux points séparés par des espaces. Les styles de TikZ sont accessibles pour les tracés.

16.1.2 Exemple avec \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}
```

16.1.3 Exemple avec \tkzDrawLines et l'option add



```
\begin{tikzpicture}[scale=.5]
 \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}
 \tkzDrawLines[add = 1 and .5,color=red](O,I O,J)
 \tkzDrawLines[add = 1 and .5,color=blue](O,i O,j)
\end{tikzpicture}
```

16.2 Multiple droites relatives au triangle \tkzDrawTLines

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

Les arguments sont une liste de trois points.

options	défaut	définition
median	median	[median](A,B,C C,A,B) médiane issue de B
altitude	median	[altitude](C,A,B) hauteur issue de A
bisector	median	[bisector](B,C,A) bissectrice issue de C
name	none	(A,B)
add= nb1 and nb2	.2 and .2	Permet de prolonger le segment

add permet de définir la longueur du trait passant par les points pt1 et pt2. Les deux nombres sont des pourcentages. Les styles de TikZ sont accessibles pour les tracés

16.3 Ajouter des labels aux droites \tkzLabelLine

\tkzLabelLine[⟨local options⟩](⟨pt1,pt2⟩){⟨label⟩}

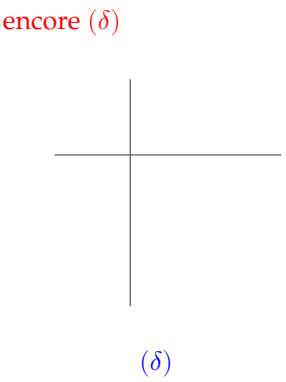
arguments	défaut	définition
label		exemple \tkzLabelLine(A,B){δ}

options	défaut	définition
pos	.5	pos est une option de TikZ mais essentielle dans ce cas

En option et en plus de **pos**, on peut utiliser tous les styles de TikZ , en particulier le placement avec **above**, **right**, ...

16.3.1 Exemple avec \tkzLabelLine

Une option importante est **pos**, c'est elle qui permet de placer le label le long de la droite. La valeur de **pos** peut être supérieure à 1 ou négative.




```
\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){encore $(\delta)$}
\end{tikzpicture}
```

17 Tacer, Marquer les segments

Il existe bien sûr, une macro pour tracer simplement un segment (il serait possible comme pour une demi-droite, de créer un style avec **\add**) .

17.1 Tracer un segment \tkzDrawSegment

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

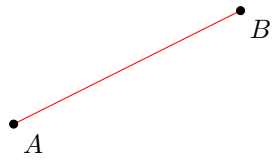
Les arguments sont une liste de deux points. Les styles de TikZ sont accessibles pour les tracés

argument	exemple	définition
(pt1,pt2)	(A,B)	trace le segment $[A,B]$

options	exemple	définition
options de TikZ		toutes les options de TikZ sont valables.

C'est bien sûr équivalent à `\draw (A)--(B);`

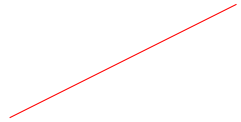
17.1.1 Exemple avec des références de points




```
\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}
```

17.1.2 Exemple avec des coordonnées

Il est préférable de référencer les points, car les points sont placées en tenant compte de **`\tkzInit`**.



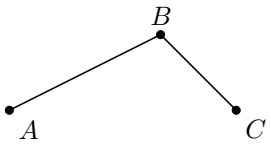
```
\begin{tikzpicture}[scale=1.5]
  \tkzDrawSegment[color=red,thin]({0,0},{2,1})
\end{tikzpicture}
```

Si les options sont les mêmes on peut tracer plusieurs segments avec la même macro.

17.2 Tracer des segments \tkzDrawSegments

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

Les arguments sont une liste de couple de deux points. Les styles de Ti*k*Z sont accessibles pour les tracés




```
\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}
```

17.3 Marquer un segment \tkzMarkSegment

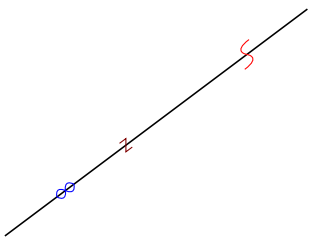
\tkzMarkSegment[⟨local options⟩](⟨pt1,pt2⟩)

La macro permet de placer une marque sur un segment.

options	défaut	définition
pos	.5	position de la marque
color	black	couleur de la marque
mark	none	choix de la marque
size	4pt	taille de la marque

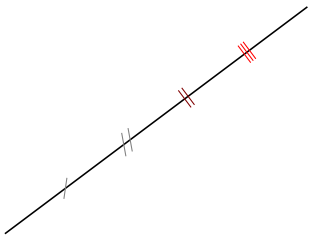
Les marques possibles sont celles fournies par Ti*k*Z, mais d'autres marques ont été crées d'après une idée de Yves Combe.

17.3.1 Marques multiples



```
\begin{tikzpicture}
  \tkzDefPoint(2,1){A}
  \tkzDefPoint(6,4){B}
  \tkzDrawSegment(A,B)
  \tkzMarkSegment[color=Maroon,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}
```

17.3.2 Utilisation de mark



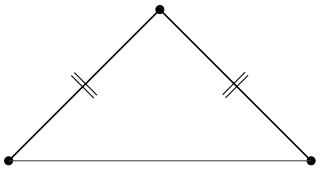
```
\begin{tikzpicture}
  \tkzDefPoint(2,1){A}
  \tkzDefPoint(6,4){B}
  \tkzDrawSegment(A,B)
  \tkzMarkSegment[color=gray,
                  pos=0.2,mark=s|](A,B)
  \tkzMarkSegment[color=gray,
                  pos=0.4,mark=s||](A,B)
  \tkzMarkSegment[color=Maroon,
                  pos=0.6,mark=||](A,B)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

17.4 Marquer des segments \tkzMarkSegments

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

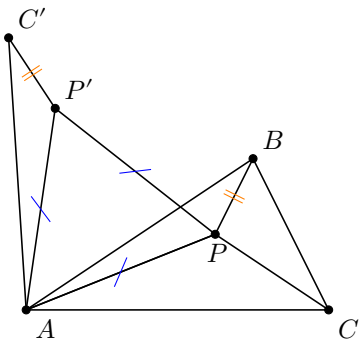
Les arguments sont une liste de couple de deux points séparés par des espaces. Les styles de Ti*k*Z sont accessibles pour les tracés.

17.4.1 Marques pour un triangle isocèle



```
\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}
```

17.5 Exemple de rotation




```
\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}
```

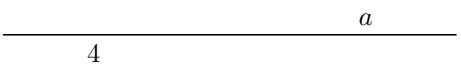
\tkzLabelSegment[⟨local options⟩](⟨pt1,pt2⟩){⟨label⟩}

Cette macro permet de placer une étiquette le long d'un segment ou encore d'une ligne. Les options sont celles de TikZ par exemple **pos**

argument	exemple	définition
label	\tkzLabelSegment(A,B){5}	texte de l'étiquette
(pt1,pt2)	(A,B)	étiquette le long de $[A,B]$

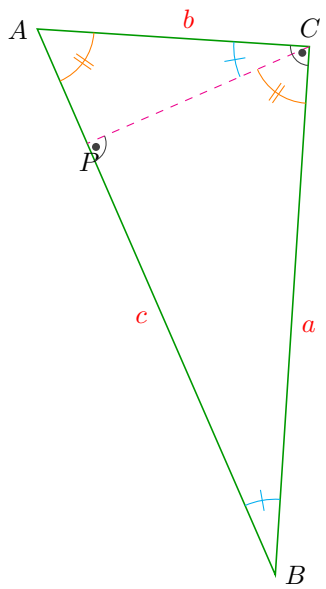
options	défaut	définition
pos	.5	position du label

17.5.1 Labels multiples




```
\begin{tikzpicture}
\tkzInit
\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}
```

17.5.2 Labels et triangle rectangle

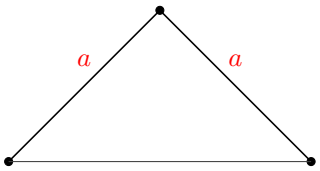



```
\begin{tikzpicture}[rotate=-60]
\tikzset{label seg style/.append style = {%
        color      = red,
        }}
\tkzDefPoint(0,1){A}
\tkzDefPoint(2,4){C}
\tkzDefPointWith[orthogonal normed,K=7](C,A)
\tkzGetPoint{B}
\tkzDrawPolygon[green!60!black](A,B,C)
\tkzDrawLine[altitude,dashed,color=magenta](B,C,A)
\tkzGetPoint{P}
\tkzLabelPoint[left](A){$A$}
\tkzLabelPoint[right](B){$B$}
\tkzLabelPoint[above](C){$C$}
\tkzLabelPoint[below](P){$P$}
\tkzLabelSegment[](B,A){$c$}
\tkzLabelSegment[swap](B,C){$a$}
\tkzLabelSegment[swap](C,A){$b$}
\tkzMarkAngles[size=1cm,
     color=cyan,mark=|](C,B,A A,C,P)
\tkzMarkAngle[size=0.75cm,
     color=orange,mark=||](P,C,B)
\tkzMarkAngle[size=0.75cm,
      color=orange,mark=||](B,A,C)
\tkzMarkRightAngles[german](A,C,B B,P,C)
\end{tikzpicture}
```

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

Les arguments sont une liste de couple de deux points. Les styles de TikZ sont accessibles pour les tracés.

17.5.3 Labels pour un triangle isocèle




```
\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}
```

18 Les triangles

18.1 Définition des triangles \tkzDefTriangle

Les macros suivantes vont permettre de définir ou de construire un triangle à partir **au moins** de deux points.

Pour le moment, il est possible de définir les triangles suivants :

- **two angles** détermine un triangle connaissant deux angles,
- **equilateral** détermine un triangle équilatéral,
- **half** détermine un triangle rectangle tel que le rapport des mesures des deux côtés adjacents à l'angle droit soit égal à 2,
- **pythagore** détermine un triangle rectangle dont les mesures des côtés sont proportionnelles à 3, 4 et 5,
- **school** détermine un triangle rectangle dont les angles sont 30, 60 et 90 degrés,
- **golden** détermine un triangle rectangle tel que le rapport des mesures des deux côtés adjacents à l'angle droit soit égal $\Phi = 1,618034$, J'ai choisi comme dénomination « triangle doré » car il rpovient du rectangle d'or et j'ai conservé la dénomination « triangle d'or » ou encore « triangle d'Euclide » pour le triangle isocèle dont les angles à la base sont de 72 degrés,
- **gold** ou **euclide** pour le triangle d'or,
- **cheops** détermine un troisième point tel que le triangle soit isocèle dont les mesures des côtés sont proportionnelles à 2, Φ et Φ.

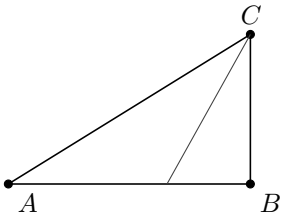
\tkzDefTriangle[⟨local options⟩](⟨A,B⟩)

les points sont ordonnés car le triangle est construit en suivant le sens direct du cercle trigonométrique. Cette macro est soit utilisée en partenariat avec **\tkzGetPoint** soit en utilisant **tkzPointResult** s'il n'est pas nécessaire de conserver le nom.

options	défaut	définition
two angles= #1 and #2	no defaut	triangle connaissant deux angles
equilateral	no defaut	triangle équilatéral
pythagore	no defaut	proportionnel au triangle de pythagore 3-4-5
school	no defaut	angles de 30, 60 et 90 degrés
gold	no defaut	angles de 72, 72 et 36 degrés, A est le sommet
euclide	no defaut	identique au précédent mais $[AB]$ est la base
golden	no defaut	rectangle en B et $AB/AC = \Phi$
cheops	no defaut	AC=BC, AC et BC sont proportionnels à 2 et Φ.

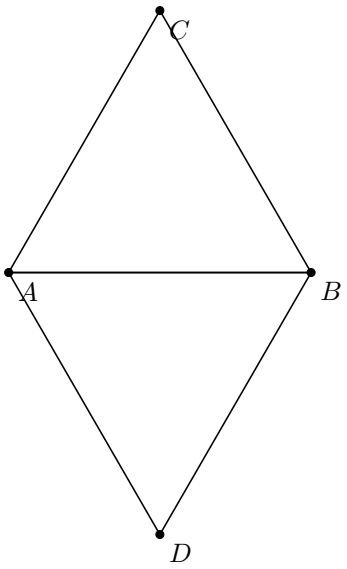
\tkzGetPoint permet de stocker le point sinon **tkzPointResult** permet une utilisation immédiate.

18.1.1 triangle doré (golden)



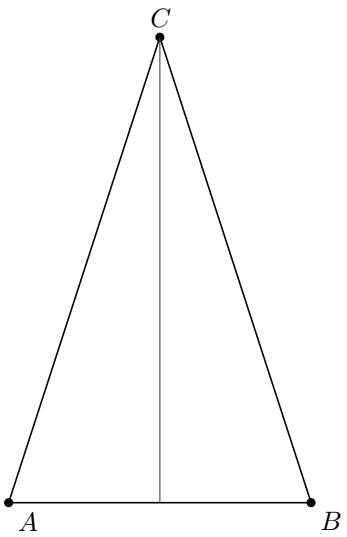

```
\begin{tikzpicture}[scale=.8]
\tkzInit[xmax=5,ymax=3] \tkzClip[space=.5]
  \tkzDefPoint(0,0){A}      \tkzDefPoint(4,0){B}
  \tkzDefTriangle[golden](A,B)\tkzGetPoint{C}
  \tkzDrawPolygon(A,B,C) \tkzDrawPoints(A,B,C)
  \tkzLabelPoints(A,B) \tkzDrawBisector(A,C,B)
  \tkzLabelPoints[above](C)
\end{tikzpicture}
```

18.1.2 triangle équilatéral




```
\begin{tikzpicture}
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(4,0){B}
  \tkzDefTriangle[equilateral](A,B)
  \tkzGetPoint{C}
  \tkzDrawPolygon(A,B,C)
  \tkzDefTriangle[equilateral](B,A)
  \tkzGetPoint{D}
  \tkzDrawPolygon(B,A,D)
  \tkzDrawPoints(A,B,C,D)
  \tkzLabelPoints(A,B,C,D)
\end{tikzpicture}
```

18.1.3 triangle d'or (euclide)




```
\begin{tikzpicture}
 \tkzDefPoint(0,0){A} \tkzDefPoint(4,0){B}
 \tkzDefTriangle[euclide](A,B)\tkzGetPoint{C}
 \tkzDrawPolygon(A,B,C)
 \tkzDrawPoints(A,B,C)
 \tkzLabelPoints(A,B)
 \tkzLabelPoints[above](C)
 \tkzDrawBisector(A,C,B)
\end{tikzpicture}
```

18.2 Tracé des triangles

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

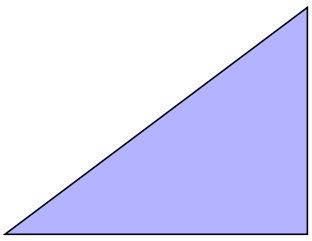
Macro semblable à la macro précédente mais les côtés sont tracés.

options	défaut	définition
two angles= #1 and #2	no defaut	triangle connaissant deux angles
equilateral	no defaut	triangle équilatéral
pythagore	no defaut	proportionnel au triangle de pythagore 3-4-5
school	no defaut	les angles sont 30, 60 et 90 degrés
gold	no defaut	les angles sont 72, 72 et 36 degrés, A est le sommet
euclide	no defaut	identique au précédent mais $[AB]$ est la base
golden	no defaut	rectangle en B et $AB/AC = \Phi$
cheops	no defaut	isocèle en C et $AC/AB = \frac{\Phi}{2}$

Dans toutes ses définitions, les dimensions du triangle dépendent des deux points de départ.

18.2.1 triangle de Pythagore

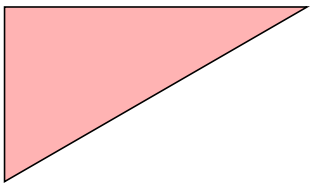
Ce triangle a des côtés dont les longueurs sont proportionnelles à 3, 4 et 5.



```
\begin{tikzpicture}
 \tkzDefPoint(0,0){A}
 \tkzDefPoint(4,0){B}
 \tkzDrawTriangle[pythagore,fill=blue!30](A,B)
\end{tikzpicture}
```

18.2.2 triangle 30 60 90 (school)

Les angles font 30, 60 et 90 degrés.



```
\begin{tikzpicture}
\tkzInit[ymin=-2.5,ymax=0,xmin=-5,xmax=0]
\tkzClip[space=.5]
  \begin{scope}[rotate=-180]
    \tkzDefPoint(0,0){A} \tkzDefPoint(4,0){B}
    \tkzDrawTriangle[school,fill=red!30](A,B)
  \end{scope}
\end{tikzpicture}
```

18.3 Tracé des médianes

L'ancienne méthode a été légèrement modifiée. À la place (A,B)(C) il faut écrire (B,C,A) C est le point qui sera lié au milieu du segment [A,B].

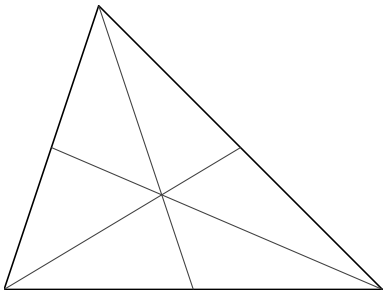
\tkzDrawMedian[⟨local options⟩](⟨point,point,point⟩)

Il y aura sans doute une autre syntaxe pour ces segments.

arguments	exemple	explication
(⟨pt1,pt2,pt3⟩)	(⟨A,B,C⟩)	[AC] est le segment cible B est le sommet

Plus dans l'esprit de la syntaxe générale, il est préférable d'utiliser : \tkzDrawLine[median](A,B,C)

18.3.1 les médianes \tkzDrawLine[median]



```
\begin{tikzpicture}[scale=1.25]
  \tkzInit[xmin=0,xmax=4,ymin=0,ymax=3] \tkzClip
  \tkzDefPoint(0,0){A} \tkzDefPoint(4,0){B}
  \tkzDefPoint(1,3){C} \tkzDrawPolygon(A,B,C)
  \tkzSetUpLine[color=blue]
  \tkzDrawLine[median](B,C,A)
  \tkzDrawLine[median](C,A,B)
  \tkzDrawLine[median](A,B,C)
\end{tikzpicture}
```

18.4 Les hauteurs \tkzDrawLine[altitude]

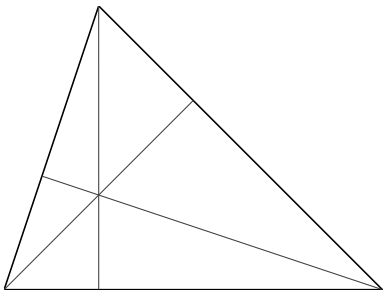
Préférable \tkzDrawLine[altitude](A,B,C)

\tkzDrawAltitude[⟨local options⟩](⟨point,point⟩)(⟨point⟩)

Il y aura sans doute une autre syntaxe pour ces segments

options	exemple	explication
(⟨pt1,pt2⟩)(⟨pt3⟩)	(⟨A,B⟩)(⟨C⟩)	[AB] est le segment cible C est le sommet

18.4.1 Exemple de hauteur



```
\begin{tikzpicture}[scale=1.25]
 \tkzInit[xmin=0,xmax=4,ymin=0,ymax=3] \tkzClip
 \tkzDefPoint(0,0){A} \tkzDefPoint(4,0){B}
 \tkzDefPoint(1,3){C} \tkzDrawPolygon(A,B,C)
 \tkzSetUpLine[color=magenta]
 \tkzDrawLine[altitude](B,C,A)
 \tkzDrawLine[altitude](C,A,B)
 \tkzDrawLine[altitude](A,B,C)
\end{tikzpicture}
```

18.5 Les bissectrices \tkzDrawLine[bisector]

Préférable \tkzDrawLine[bisector](A,B,C)

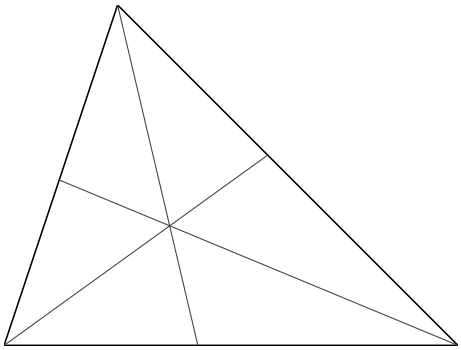
\tkzDrawBisector[⟨local options⟩](⟨point,point⟩)(⟨point⟩)

Il faut donner l'angle dans le sens direct

options	exemple	explication
(⟨pt1,pt2,pt3⟩)	(⟨A,B,C⟩)	Le sommet est B

18.5.1 Bissectrices dans un triangle

Il faut donner les angles dans le sens direct.



```
\begin{tikzpicture}[scale=1.5]
 \tkzInit[xmin=0,xmax=4,ymin=0,ymax=3] \tkzClip
 \tkzDefPoint(0,0){A} \tkzDefPoint(4,0){B}
 \tkzDefPoint(1,3){C} \tkzDrawPolygon(A,B,C)
 \tkzSetUpLine[color=purple]
 \tkzDrawLine[bisector](B,C,A)
 \tkzDrawLine[bisector](C,A,B)
 \tkzDrawLine[bisector](A,B,C)
\end{tikzpicture}
```

19 Triangles spécifiques avec \tkzDefSpcTriangle

Les centres de certains triangles ont été définis dans la section "points", ici il s'agit de déterminer les trois sommets de triangles spécifiques.

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

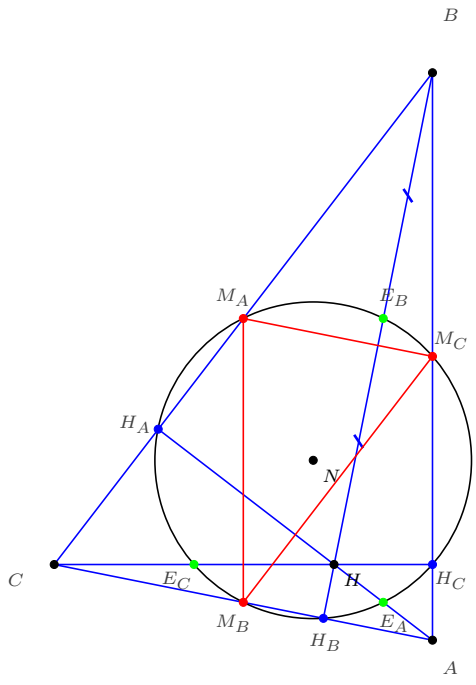
L'ordre des points est important !

options	défaut	définition
in or incentral	centroid	triangle connaissant deux angles
ex or excentral	centroid	triangle équilatéral
extouch	centroid	proportionnel au triangle de pythagore 3-4-5
intouch or contact	centroid	angles de 30, 60 et 90 degrés
centroid or medial	centroid	angles de 72, 72 et 36 degrés, A est le sommet
orthic	centroid	identique au précédent mais $[AB]$ est la base
feuerbach	centroid	rectangle en B et $AB/AC = \Phi$
euler	centroid	AC=BC, AC et BC sont proportionnels à 2 et Φ.
tangential	centroid	AC=BC, AC et BC sont proportionnels à 2 et Φ.
name	no defaut	AC=BC, AC et BC sont proportionnels à 2 et Φ.

\tkzGetPoint permet de stocker le point sinon **tkzPointResult** permet une utilisation immédiate.

19.0.1 \tkzDefSpcTriangle option "medial" ou "centroid"

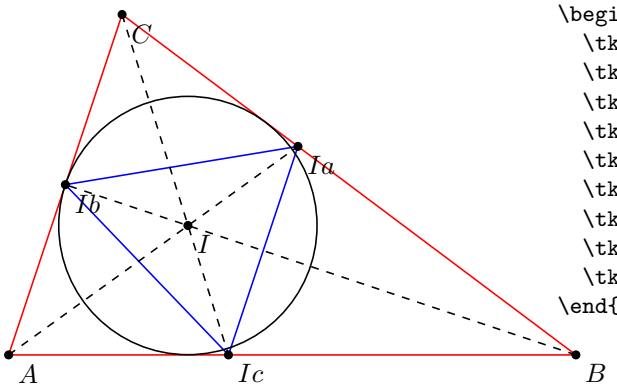
Il est défini par les intersections des médianes avec les côtés d'un triangle. On obtient dans l'exemple suivant le cercle d'Euler qui passe par les points précédemment définis.




```
\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[color=blue](A,B,C)
 \tkzDrawCircle(N,E_A)
 \tkzDrawSegments[blue](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=red](M_A,M_B,M_C)
\end{tikzpicture}
```

19.0.2 \tkzDefSpcTriangle option : "in" ou "incentral"

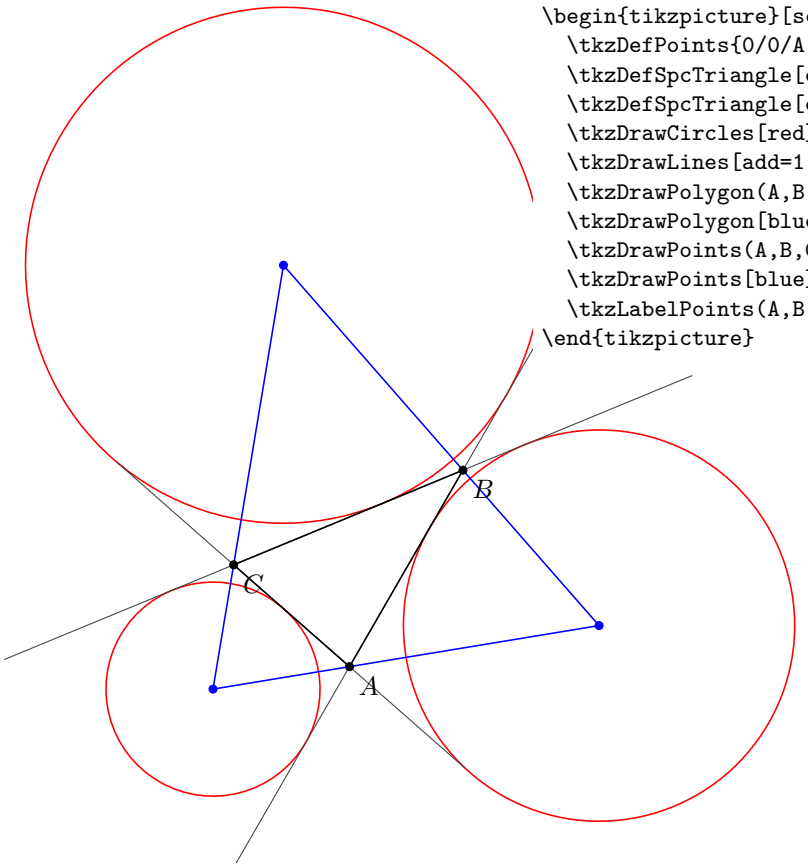
On obtient les intersections des bissectrices avec les côtés




```
\begin{tikzpicture}[scale=1.5]
  \tkzDefPoints{ 0/0/A,5/0/B,1/3/C}
  \tkzDefSpcTriangle[in,name=I](A,B,C){a,b,c}
  \tkzInCenter(A,B,C)\tkzGetPoint{I}
  \tkzDrawPolygon[red](A,B,C)
  \tkzDrawPolygon[blue](Ia,Ib,Ic)
  \tkzDrawPoints(A,B,C,I,Ia,Ib,Ic)
  \tkzLabelPoints(A,B,C,I,Ia,Ib,Ic)
  \tkzDrawCircle[in](A,B,C)
  \tkzDrawSegments[dashed](A,Ia B,Ib C,Ic)
\end{tikzpicture}
```

19.0.3 \tkzDefSpcTriangle option : "extouch"

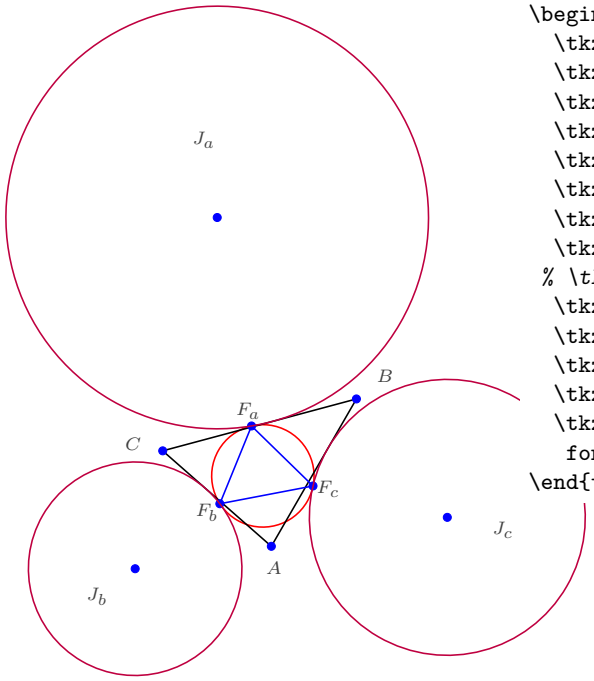
On obtient les les poits de contact des cercles exinscrits ainsi que les triangle formé par les centres des cercles exinscrits.




```
\begin{tikzpicture}[scale=.5,rotate=60]
  \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}
  \tkzDrawCircles[red](Ja,Ta Jb,Tb Jc,Tc)
  \tkzDrawLines[add=1 and 1](A,B B,C C,A)
  \tkzDrawPolygon(A,B,C)
  \tkzDrawPolygon[blue](Ja,Jb,Jc)
  \tkzDrawPoints(A,B,C)
  \tkzDrawPoints[blue](Ja,Jb,Jc)
  \tkzLabelPoints(A,B,C)
\end{tikzpicture}
```

19.0.4 \tkzDefSpcTriangle Triangle "feuerbach"

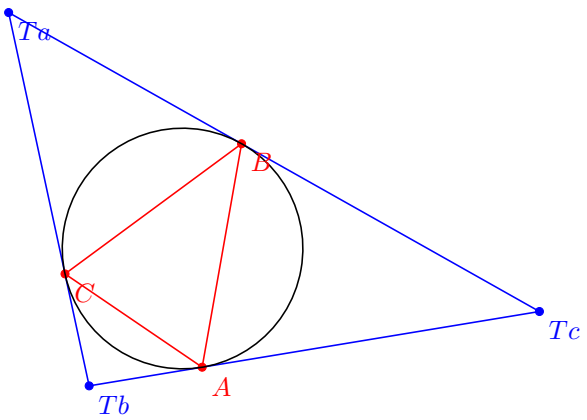
Le cercle d'Euler est tangent ici aux cercles exinscrits. Les points de tangences définissent le triangle de Feuerbach.




```
\begin{tikzpicture}[scale=.75,rotate=60]
  \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}
  \tkzDrawCircle[red](N,F_a)
 % \tkzDrawCircle[in](A,B,C)
  \tkzDrawPolygon(A,B,C)
  \tkzDrawPolygon[blue](F_a,F_b,F_c)
  \tkzDrawPoints[blue](J_a,J_b,J_c,F_a,F_b,F_c,A,B,C))
  \tkzDrawCircles[purple](J_a,F_a J_b,F_b J_c,F_c)
  \tkzAutoLabelPoints[center=N,dist=.3,
   font=\scriptsize](A,B,C,F_a,F_b,F_c,J_a,J_b,J_c)
\end{tikzpicture}
```

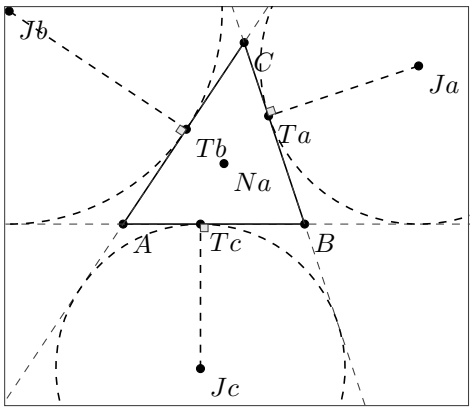
19.0.5 \tkzDefSpcTriangle Triangle "tangential"

Triangle formé par les tangentes aux somments du cercle circonscrits



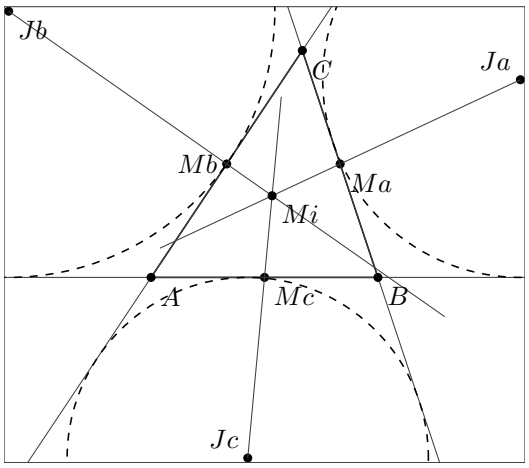

```
\begin{tikzpicture}[scale=.5,rotate=80]
  \tkzDefPoints{0/0/A,6/0/B,1.8/4/C}
  \tkzDefSpcTriangle[tangential,
    name=T](A,B,C){a,b,c}
  \tkzDrawPolygon[red](A,B,C)
  \tkzDrawPolygon[blue](Ta,Tb,Tc)
  \tkzDrawPoints[red](A,B,C)
  \tkzDrawPoints[blue](Ta,Tb,Tc)
  \tkzDefCircle[circum](A,B,C)
  \tkzGetPoint{O}
  \tkzDrawCircle(O,A)
  \tkzLabelPoints[red](A,B,C)
  \tkzLabelPoints[blue](Ta,Tb,Tc)
\end{tikzpicture}
```

19.0.6 Exemple avec le point de Nagel




```
\begin{tikzpicture}[scale=.4]
 \tkzDefPoints{% x   y   name
               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}
\tkzDrawPoints(Ja,Jb,Jc,Ta,Tb,Tc)
\tkzLabelPoints(Ja,Jb,Jc,Ta,Tb,Tc)
\tkzDrawPolygon[](A,B,C)
\tkzDefTriangleCenter[nagel](A,B,C)
\tkzGetPoint{Na}
\tkzDrawPoints(B,C,A,Na)
\tkzLabelPoints(B,C,A,Na)
\tkzShowBB\tkzClipBB
\tkzDrawLines[add=1 and 1,dashed](A,B B,C C,A)
\tkzDrawCircles[dashed](Ja,Ta Jb,Tb Jc,Tc)
\tkzDrawSegments[dashed](Ja,Ta Jb,Tb Jc,Tc)
\tkzMarkRightAngles[fill=gray!20](Ja,Ta,C
                Jb,Tb,A Jc,Tc,B)
\end{tikzpicture}
```

19.0.7 Exemple avec le mittenpunkt




```
\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(Ma,Mb,Mc,Ja,Jb,Jc)
 \tkzClipBB
 \tkzDrawPolygon[](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[dashed](Ja,Ta Jb,Tb Jc,Tc)
 \tkzDrawPoints(B,C,A,Mi)
 \tkzLabelPoints(B,C,A,Mi)
 \tkzLabelPoints[left](Mb)
 \tkzLabelPoints(Ma,Mc,Jb,Jc)
 \tkzLabelPoints[above left](Ja,Jc)
 \tkzShowBB
\end{tikzpicture}
```

20 Définition de polygones

20.1 Définir les points d'un carré

Nous avons vu les définitions de certains triangles. Voyons celles de certains quadrilatères et des polygones réguliers.

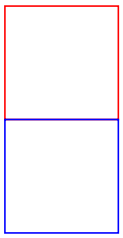
\tkzDefSquare(⟨pt1,pt2⟩)

Le carré est défini dans le sens direct. À partir de deux points, on obtient deux autres points tel que les quatre pris dans l'ordre forme un carré. Le carré est défini dans le sens direct. Les résultats sont dans **tkzFirstPointResult** et **tkzSecondPointResult**.
On peut les renommer avec **\tkzGetPoints**

Arguments	exemple	explication
(⟨pt1,pt2⟩)	**\tkzDefSquare**(⟨A,B⟩)	Le carré est défini dans le sens direct

20.1.1 Utilisation de \tkzDefSquare avec deux points

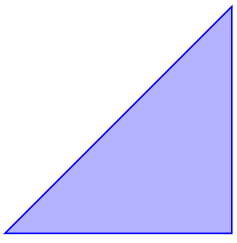
Il faut remarquer l'inversion des deux premiers points et le résultat.



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

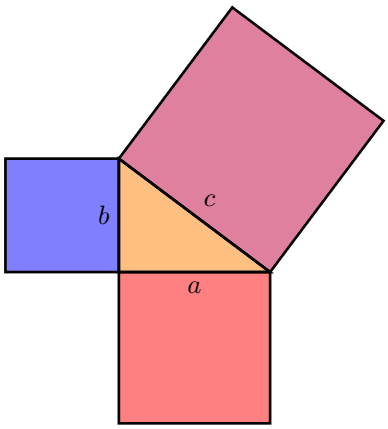
On peut n'avoir besoin que d'un point pour tracer un triangle isocèle rectangle alors on utilise **\tkzGetFirstPoint** ou **\tkzGetSecondPoint**

20.1.2 Utilisation de \tkzDefSquare pour obtenir un triangle isocèle rectangle



```
\begin{tikzpicture}[scale=1]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(3,0){B}
  \tkzDefSquare(A,B) \tkzGetFirstPoint{C}
  \tkzDrawPolygon[color=blue,fill=blue!30](A,B,C)
\end{tikzpicture}
```

20.1.3 Théorème de Pythagore et \tkzDefSquare




```
\begin{tikzpicture}[scale=.5]
\tkzInit
\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[fill = red!50 ](A,C,G,H)
\tkzFillPolygon[fill = blue!50 ](C,B,I,J)
\tkzFillPolygon[fill = purple!50](B,A,E,F)
\tkzFillPolygon[fill = orange,opacity=.5](A,B,C)
\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[](A,C){$a$}
\tkzLabelSegment[](C,B){$b$}
\tkzLabelSegment[swap](A,B){$c$}
\end{tikzpicture}
```

20.2 Définition du parallélogramme

20.3 Définir les points d'un parallélogramme

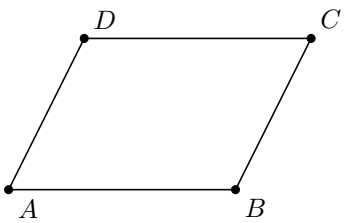
Il s'agit de compléter trois points afin d'obtenir un parallélogramme.

\tkzDefParallelogram(⟨pt1,pt2,pt3⟩)

À partir de trois points, on obtient un autre point tel que les quatre pris dans l'ordre forme un parallélogramme. Le résultat est dans **tkzPointResult**.
On peut le renommer avec **\tkzGetPoint**

arguments	défaut	définition
(⟨pt1,pt2,pt3⟩)	no default	Trois points sont nécessaires

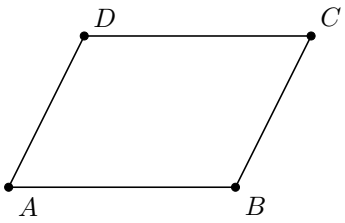
20.3.1 Exemple de définition d'un parallélogramme




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

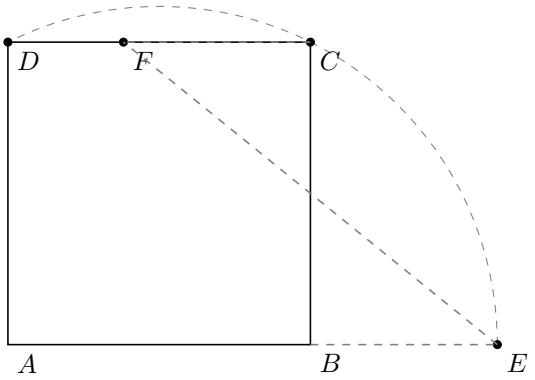
20.3.2 Exemple simple avec \colinear= at

Explication de la définition d'un parallélogramme




```
\begin{tikzpicture}[scale=1]
  \tkzDefPoints{0/0/A,3/0/B,4/2/C}
  \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}
```

20.3.3 Construction du rectangle d'or avec \colinear= at




```
\begin{tikzpicture}[scale=.5]
  \tkzInit[xmax=14,ymax=10]
  \tkzClip[space=1]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(8,0){B}
  \tkzDefMidPoint(A,B)\tkzGetPoint{I}
  \tkzDefSquare(A,B)\tkzGetPoints{C}{D}
  \tkzDrawSquare(A,B)
  \tkzInterLC(A,B)(I,C)\tkzGetPoints{G}{E}
  \tkzDrawArc[style=dashed,color=gray](I,E)(D)
  \tkzDefPointWith[colinear= at C](E,B)
  \tkzGetPoint{F}
  \tkzDrawPoints(C,D,E,F)
  \tkzLabelPoints(A,B,C,D,E,F)
  \tkzDrawSegments[style=dashed,color=gray]%
(E,F C,F B,E)
\end{tikzpicture}
```

20.4 Tracé un carré

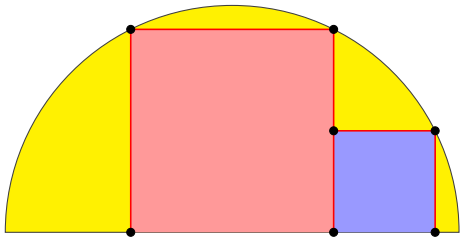
\tkzDrawSquare[⟨local options⟩](⟨pt1,pt2⟩)

La macro trace un carré mais pas les sommets. Il est possible de colorier l'intérieur. L'ordre des points est celui du sens direct du cercle trigonométrique

arguments	exemple	explication
(⟨pt1,pt2⟩)	**\tkzDrawSquare**(⟨A,B⟩)	**\tkzGetPoints{C}{D}**

options	exemple	explication
Options TikZ	red,line width=1pt	

20.4.1 Il s'agit d'inscrire deux carrés dans un demi-cercle.



```
\begin{tikzpicture}[scale=.75]
   \tkzInit[ymax=8,xmax=8]
 \tkzClip[space=.25]    \tkzDefPoint(0,0){A}
 \tkzDefPoint(8,0){B}  \tkzDefPoint(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[fill=yellow](I,B)(A)
 \tkzFillPolygon[color=red!40](H,E,F,G)
 \tkzFillPolygon[color=blue!40](H,J,K,L)
 \tkzDrawPolySeg[color=red](H,E,F,G)
 \tkzDrawPolySeg[color=red](J,K,L)
 \tkzDrawPoints(E,G,H,F,J,K,L)
\end{tikzpicture}
```

20.5 Le rectangle d'or

\tkzDefGoldRectangle(⟨point,point⟩)

La macro détermine un rectangle dont le rapport des dimensions est le nombre Φ. Les points créés sont dans **tkzFirstPointResult** et **tkzSecondPointResult**. On peut les obtenir avec la macro **\tkzGetPoints**. La macro suivante permet de tracer le rectangle.

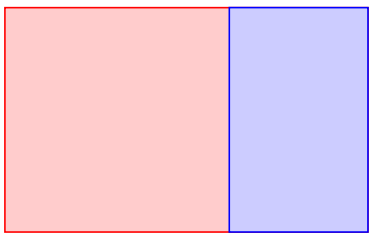
arguments	exemple	explication
(⟨pt1,pt2⟩)	(⟨A,B⟩)	Si C et D sont créés alors $AB/BC = \Phi$

\tkzDrawGoldRectangle[⟨local options⟩](⟨point,point⟩)

arguments	exemple	explication
(⟨pt1,pt2⟩)	(⟨A,B⟩)	Trace le rectangle d'or basé sur le segment $[AB]$

options	exemple	explication
Options TikZ	red,line width=1pt	

20.5.1 Rectangles d'or



```
\begin{tikzpicture}[scale=.6]
 \tkzDefPoint(0,0){A}      \tkzDefPoint(8,0){B}
 \tkzDefGoldRectangle(A,B) \tkzGetPoints{C}{D}
 \tkzDefGoldRectangle(B,C) \tkzGetPoints{E}{F}
 \tkzDrawPolygon[color=red,fill=red!20](A,B,C,D)
 \tkzDrawPolygon[color=blue,fill=blue!20](B,C,E,F)
\end{tikzpicture}
```

20.6 Tracer un polygone

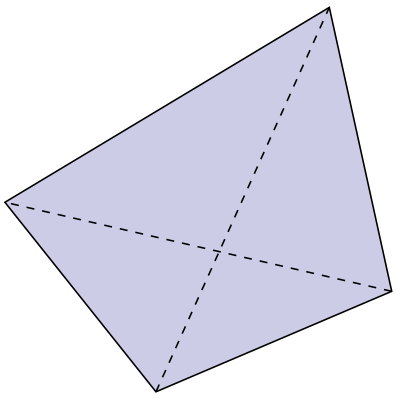
\tkzDrawPolygon[⟨local options⟩](⟨liste de points⟩)

Il suffit de donner une liste de points et la macro trace le polygone en utilisant les options de Ti*k*Z présentes.

arguments	exemple	explication
(⟨pt1,pt2,pt3,...⟩)	\tkzDrawPolygon[gray,dashed](A,B,C)	Tracé d'un triangle

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

20.6.1 Tracer un polygone 1



```
\begin{tikzpicture} [rotate=18,scale=1.5]
 \tkzDefPoint(0,0){A}
 \tkzDefPoint(2.25,0.2){B}
 \tkzDefPoint(2.5,2.75){C}
 \tkzDefPoint(-0.75,2){D}
 \tkzDrawPolygon[fill=black!50!blue!20!](A,B,C,D)
 \tkzDrawSegments[style=dashed](A,C B,D)
\end{tikzpicture}
```

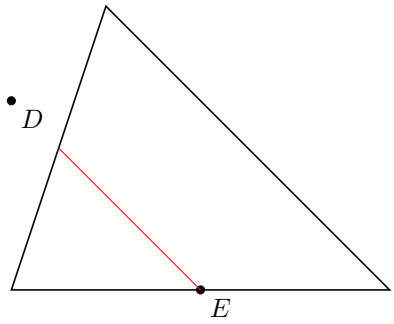
20.7 Clipper un polygone

\tkzClipPolygon[⟨local options⟩](⟨liste de points⟩)

Cette macro permet de contenir les différentes tracés dans le polygone désigné.

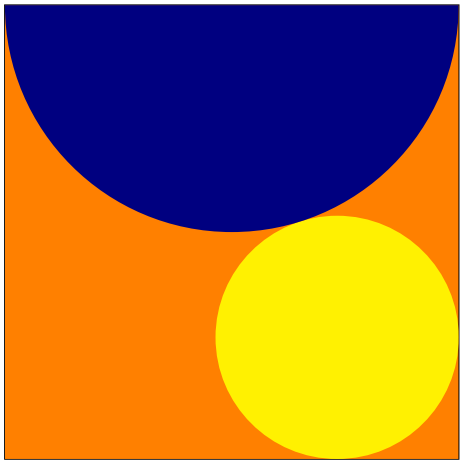
options	exemple	explication
(⟨pt1,pt2⟩)	(⟨A,B⟩)	

20.7.1 Exemple simple avec \tkzClipPolygon




```
\begin{tikzpicture}[scale=1.25]
 \tkzInit[xmin=0,xmax=4,ymin=0,ymax=3]
 \tkzClip[space=.5]
 \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[color=red](D,E)
\end{tikzpicture}
```

20.7.2 Exemple Sangaku dans un carré



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

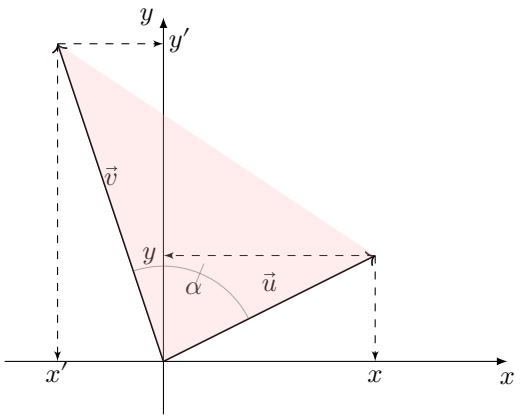
20.8 Colorier un polygone

\tkzFillPolygon[⟨local options⟩](⟨liste de points⟩)

On peut colorier en traçant le polygone mais là on colorie l'intrieur du polygone sans le tracer.

options	exemple	explication
(⟨pt1,pt2,…⟩)	(⟨A,B,…⟩)	

20.8.1 Colorier un polygone




```
\begin{tikzpicture}[scale=0.7]
\tkzInit[xmin=-3,xmax=6,ymin=-1,ymax=6]
\tkzDrawX[noticks]
\tkzDrawY[noticks]
\tkzDefPoint(0,0){O}  \tkzDefPoint(4,2){A}
\tkzDefPoint(-2,6){B}
\tkzPointShowCoord[xlabel=$x$,ylabel=$y$](A)
\tkzPointShowCoord[xlabel=$x'$,ylabel=$y'$,%
                   ystyle={right=2pt}](B)
\tkzDrawSegments[->](O,A O,B)
\tkzLabelSegment[above=3pt](O,A){$\vec{u}$}
\tkzLabelSegment[above=3pt](O,B){$\vec{v}$}
\tkzMarkAngle[fill= yellow,size=1.8cm,%
              opacity=.5](A,O,B)
\tkzFillPolygon[red!30,opacity=0.25](A,B,O)
\tkzLabelAngle[pos = 1.5](A,O,B){$\alpha$}
\end{tikzpicture}
```

21 Les Cercles

Parmi les macros suivantes, l'une va permettre de tracer un cercle, ce qui n'est pas un réel exploit. Pour cela, il va falloir connaître le centre du cercle et soit le rayon du cercle, soit un point de la circonférence. Il m'a semblé que l'utilisation la plus fréquente était de tracer un cercle de centre donné passant par un point donné. Ce sera la méthode par défaut, sinon il faudra utiliser l'option **R**. Il existe un grand nombre de cercles particuliers, par exemple le cercle circonscrit à un triangle.

- J'ai créé une première macro **\tkzDefCircle** qui permet en fonction d'un cercle particulier de récupérer son centre et la mesure du rayon en cm. Cette récupération se fait avec les macros **\tkzGetPoint** et **\tkzGetLength**,
- ensuite une macro **\tkzDrawCircle**,
- puis une macro qui permet de colorier un disque, mais sans tracer le cercle **\tkzFillCircle**,
- parfois, il est nécessaire qu'un dessin soit contenu dans un disque c'est le rôle attribuer à **\tkzClipCircle**,
- Il reste enfin à pouvoir donner un label pour désigner un cercle et si plusieurs possibilités sont offertes, nous verrons ici **\tkzLabelCircle**.

21.1 Caractéristiques d'un cercle : \tkzDefCircle

Cette macro permet de récupérer les caractéristiques (centre et rayon) de certains cercles.

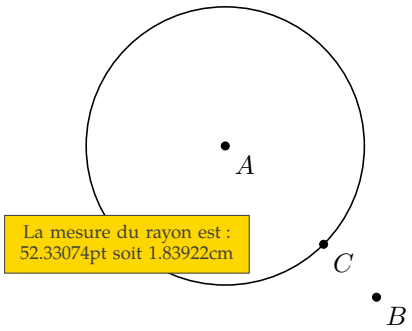
\tkzDefCircle[⟨local options⟩](⟨A,B⟩) ou (⟨A,B,C⟩)

Attention les arguments sont des listes de deux ou bien de trois points. Cette macro est, soit utilisée en partenariat avec **\tkzGetPoint** et/ou **\tkzGetLength** pour obtenir le centre et le rayon du cercle, soit en utilisant **tkzPointResult** et **tkzLengthResult** s'il n'est pas nécessaire de conserver les résultats.

options	défaut	définition
through	through	cercle caractérisé par deux points définissant un rayon
diameter	through	cercle caractérisé par deux points définissant un diamètre
circum	through	cercle circonscrit à un triangle
in	through	cercle inscrit dans à un triangle
ex	through	cercle exinscrit à un triangle
euler or nine	through	Cercle d'Euler
spieker	through	Cercle de Spieker
apollonius	through	Cercle d'Apollonius
orthogonal	through	Cercle de centre donné orthogonal à un autre cercle
orthogonal through	through	Cercle orthogonal à un autre cercle passant par deux points
K	1	Coefficient utilisé pour un cercle d'Apollonius

Dans les exemples suivants, je trace les cercles avec une macro pas encore présentée, mais ce n'est pas nécessaire. Dans certains cas on peut seulement avoir besoin du centre ou encore du rayon.

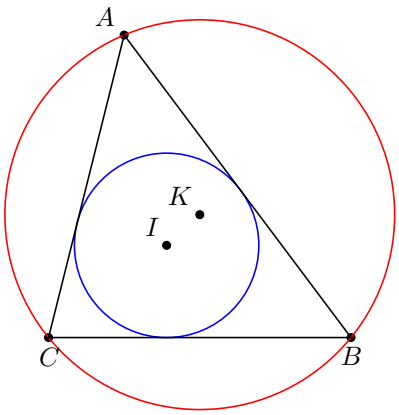
21.1.1 Exemple avec un point aléatoire




```
\begin{tikzpicture}[scale=1]
\tkzDefPoint(0,4){A}
\tkzDefPoint(2,2){B}
\tkzDefMidPoint(A,B) \tkzGetPoint{I}
\tkzDefRandPointOn[segment = I--B]
 \tkzGetPoint{C}
\tkzDefCircle[through](A,C)
\tkzGetLength{rACpt}
\tkzpttocm(\rACpt){rACcm}
\tkzDrawCircle(A,C)
\tkzDrawPoints(A,B,C)
\tkzLabelPoints(A,B,C)
\tkzLabelCircle[draw,fill=Gold,%
text width=3cm,text centered,
font=\scriptsize](A,C)(-90)%
{La mesure du rayon est :
 \rACpt pt soit \rACcm cm}
\end{tikzpicture}
```

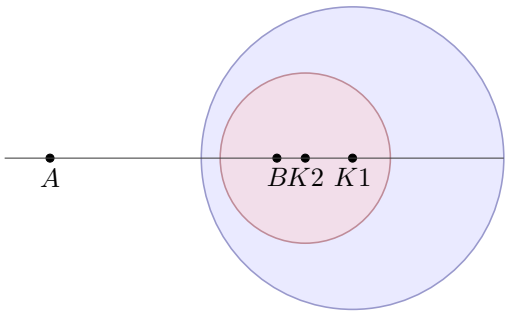
21.1.2 Cercles inscrit et circonscrit pour un triangle donné

Vous pouvez également obtenir le centre du cercle inscrit et la projection de celui-ci sur un côté du triangle avec **\tkzGetFirstPointI** et **\tkzGetSecondPointIb**.




```
\begin{tikzpicture}[scale=1]
   \tkzDefPoint(2,2){A}
   \tkzDefPoint(5,-2){B}
   \tkzDefPoint(1,-2){C}
   \tkzDefCircle[in](A,B,C)
   \tkzGetPoint{I}    \tkzGetLength{rIN}
   \tkzDefCircle[circum](A,B,C)
   \tkzGetPoint{K}   \tkzGetLength{rCI}
   \tkzDrawPoints(A,B,C,I,K)
   \tkzDrawCircle[R,blue](I,\rIN pt)
   \tkzDrawCircle[R,red](K,\rCI pt)
   \tkzLabelPoints[below](B,C)
   \tkzLabelPoints[above left](A,I,K)
   \tkzDrawPolygon(A,B,C)
\end{tikzpicture}
```

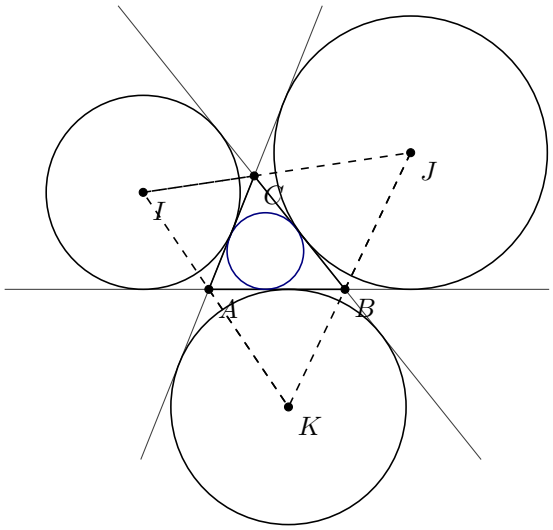
21.1.3 Cercles d'Apollonius colorié pour un segment donné




```
\begin{tikzpicture}[scale=0.75]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(4,0){B}
  \tkzDefCircle[apollonius,K=2](A,B)
  \tkzGetPoint{K1}
  \tkzGetLength{rAp}
  \tkzDrawCircle[R,color = blue!50!black,
      fill=blue!20,opacity=.4](K1,\rAp pt)
  \tkzDefCircle[apollonius,K=3](A,B)
  \tkzGetPoint{K2}   \tkzGetLength{rAp}
  \tkzDrawCircle[R,color=red!50!black,
   fill=red!20,opacity=.4](K2,\rAp pt)
  \tkzLabelPoints[below](A,B,K1,K2)
  \tkzDrawPoints(A,B,K1,K2)
  \tkzDrawLine[add=.2 and 1](A,B)
\end{tikzpicture}
```

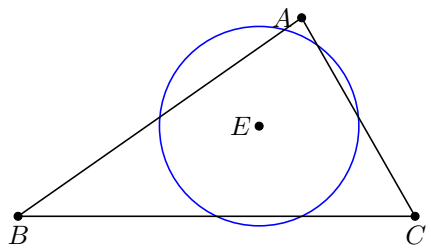
21.1.4 Cercles exinscrits à un triangle donné

Vous pouvez également obtenir le centre et la projection de celui-ci sur un côté du triangle avec **\tkzGetFirstPoint{Jb}** et **\tkzGetSecondPoint{Tb}**.



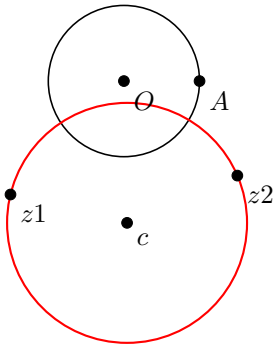

```
\begin{tikzpicture}[scale=.6]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(3,0){B}
  \tkzDefPoint(1,2.5){C}
  \tkzDefCircle[ex](A,B,C) \tkzGetPoint{I}
    \tkzGetLength{rI}
  \tkzDefCircle[ex](C,A,B) \tkzGetPoint{J}
    \tkzGetLength{rJ}
  \tkzDefCircle[ex](B,C,A) \tkzGetPoint{K}
    \tkzGetLength{rK}
   \tkzDefCircle[in](B,C,A) \tkzGetPoint{O}
     \tkzGetLength{rO}
  \tkzDrawLines[add=1.5 and 1.5](A,B A,C B,C)
  \tkzDrawPoints(I,J,K)
  \tkzDrawPolygon(A,B,C)
  \tkzDrawPolygon[dashed](I,J,K)
  \tkzDrawCircle[R,blue!50!black](O,\rO)
  \tkzDrawSegments[dashed](A,K B,J C,I)
  \tkzDrawPoints(A,B,C)
  \tkzDrawCircles[R](J,{\rJ} I,{\rI} K,{\rK})
  \tkzLabelPoints(A,B,C,I,J,K)
\end{tikzpicture}
```

21.1.5 Cercle d'Euler pour un triangle donné



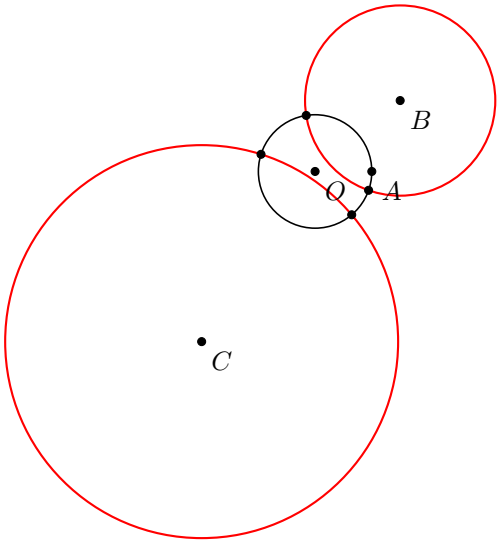

```
\begin{tikzpicture}[scale=.75]
   \tkzDefPoint(5,3.5){A}
   \tkzDefPoint(0,0){B} \tkzDefPoint(7,0){C}
   \tkzDefCircle[euler](A,B,C)
   \tkzGetPoint{E}  \tkzGetLength{rEuler}
   \tkzDrawPoints(A,B,C,E)
   \tkzDrawCircle[R,blue](E,\rEuler pt)
   \tkzDrawPolygon(A,B,C)
   \tkzLabelPoints[below](B,C)
   \tkzLabelPoints[left](A,E)
\end{tikzpicture}
```

21.1.6 Cercle orthogonal passant par deux points donnés




```
\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[thick,color=red](tkzPointResult,z1)
  \tkzDrawPoints[fill=red,color=black,
  size=4](O,A,z1,z2,c)
  \tkzLabelPoints(O,A,z1,z2,c)
\end{tikzpicture}
```

21.1.7 Cercle orthogonal de centre donné




```
\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)
  \tkzDrawCircle[thick,color=red](B,z1)
  \tkzDrawCircle[thick,color=red](C,t1)
  \tkzDrawPoints(t1,t2,C)
  \tkzDrawPoints(z1,z2,O,A,B)
  \tkzLabelPoints(O,A,B,C)
\end{tikzpicture}
```

21.2 Tangente à un cercle

Deux constructions sont proposées. La première est la construction d'une tangente à un cercle en un point donné de ce cercle et la seconde est la construction d'une tangente à un cercle passant par un point donné hors d'un disque.

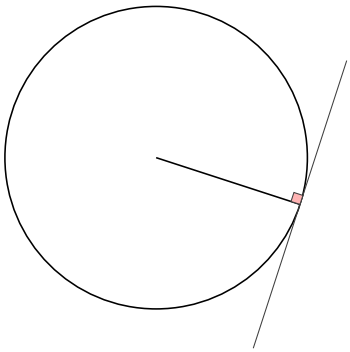
\tkzDefTangent[⟨local options⟩](⟨pt1,pt2⟩) ou (⟨pt1,dim⟩)

Le paramètre entre parenthèses est le centre du cercle ou bien le centre du cercle et un point du cercle ou encore le centre et le rayon.

options	défaut	définition
at=pt	at	tangente en un point du cercle
from=pt	at	tangente à un cercle passant par un point
from with R=pt	at	idem, mais le cercle est défini par centre+rayon

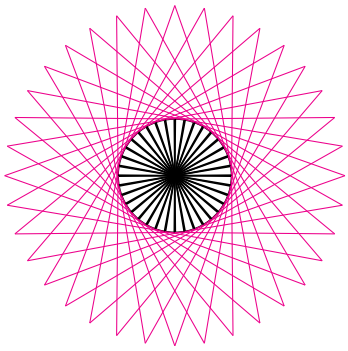
La tangente n'est pas tracée.Un second point de celle-ci est donné par **tkzPointResult**.

21.2.1 Exemple de tangente passant par un point du cercle



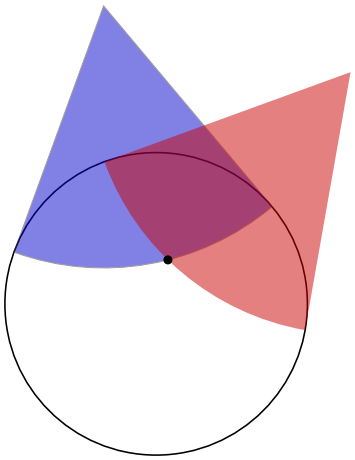
```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(6,6){E}
  \tkzDefRandPointOn[circle=center O radius 4cm]
  \tkzGetPoint{A}
  \tkzDrawSegment(O,A)
  \tkzDrawCircle(O,A)
  \tkzDefTangent[at=A](O)
  \tkzGetPoint{h}
  \tkzDrawLine[add = 4 and 3](A,h)
  \tkzMarkRightAngle[fill=red!30](O,A,h)
\end{tikzpicture}
```

21.2.2 Exemple de tangentes passant par un point extérieur



```
\begin{tikzpicture}[scale=0.75]
  \tkzDefPoint(3,3){c}
  \tkzDefPoint(6,3){a0}
  \tkzRadius=1 cm
  \tkzDrawCircle[R](c,\tkzRadius)
  \foreach \an in {0,10,...,350}{
     \tkzDefPointBy[rotation=center c angle \an](a0)
     \tkzGetPoint{a}
     \tkzDefTangent[from with R = a](c,\tkzRadius)
     \tkzGetPoints{e}{f}
     \tkzDrawLines[color=magenta](a,f a,e)
      \tkzDrawSegments(c,e c,f)
      }%
\end{tikzpicture}
```

21.2.3 Exemple d'Andrew Mertz



```
\begin{tikzpicture}[scale=.5]
\tkzDefPoint(100:8){A}\tkzDefPoint(50:8){B}
\tkzDefPoint(0,0){C} \tkzDefPoint(0,4){R}
\tkzDrawCircle(C,R)
\tkzDefTangent[from = A](C,R)  \tkzGetPoints{D}{E}
\tkzDefTangent[from = B](C,R)  \tkzGetPoints{F}{G}
\tkzDrawSector[fill=blue!80!black,opacity=0.5](A,D)(E)
\tkzFillSector[color=red!80!black,opacity=0.5](B,F)(G)
\tkzInterCC(A,D)(B,F) \tkzGetSecondPoint{I}
\tkzDrawPoint[color=black](I)
\end{tikzpicture}
```

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

22 Tracer, étiqueter Les Cercles

Parmi les macros suivantes, l'une va permettre de tracer un cercle, ce qui n'est pas un réel exploit. Pour cela, il va falloir connaître le centre du cercle et soit le rayon du cercle, soit un point de la circonférence. Il m'a semblé que l'utilisation la plus fréquente était de tracer un cercle de centre donné passant par un point donné. Ce sera la méthode par défaut, sinon il faudra utiliser l'option **R**.

— J'ai créé une première macro **\tkzDrawCircle**,
— puis une macro qui permet de colorier un disque, mais sans tracer le cercle **\tkzFillCircle**,
— parfois, il est nécessaire qu'un dessin soit contenu dans un disque c'est le rôle attribuer à **\tkzClipCircle**,
— Il reste enfin à pouvoir donner un label pour désigner un cercle et si plusieurs possibilités sont offertes, nous verrons ici **\tkzLabelCircle**.

22.1 Tracer un cercle

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

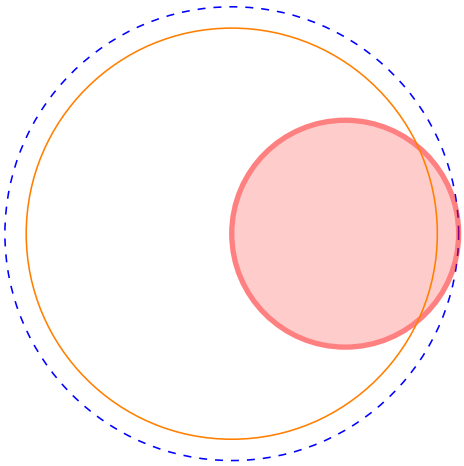
Attention les arguments sont des listes de deux ou bien de trois points. Les cercles que l'on peut tracer sont les mêmes que pour la macro précédente. Une option supplémentaire **R** afin de donner directement une mesure.

options	défaut	définition
through	through	cercle avec deux points définissant un rayon
diameter	through	cercle avec deux points définissant un diamètre
R	through	cercle caractérisé par un point et la mesure d'un rayon

Il faut ajouter bien sûr tous les styles de Ti*k*Zpour les tracés

22.1.1 Cercles et styles, tracer un cercle et colorier le disque

On va voir qu'il est possible de colorier un disque, tout en traçant le cercle.



```
\begin{tikzpicture}
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(3,0){A}
 % cercle de centre O et passant par A
  \tkzDrawCircle[color=blue,style=dashed](O,A)
 % cercle de diamètre $[OA]$
  \tkzDrawCircle[diameter,color=red,%
                 line width=2pt,fill=red!40,%
                 opacity=.5](O,A)
 % cercle de centre O et de rayon = exp(1) cm
  \edef\rayon{\fpeval{exp(1)}}
  \tkzDrawCircle[R,color=orange](O,\rayon cm)
\end{tikzpicture}
```

22.2 Tracer des cercles

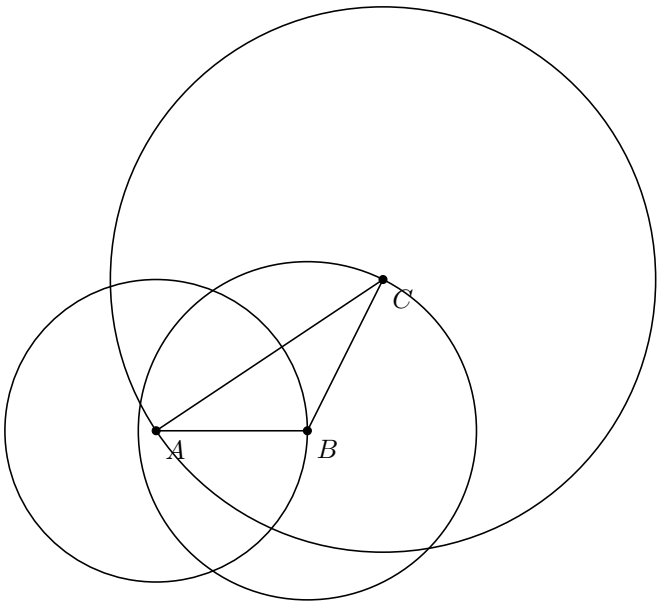
`\tkzDrawCircles[⟨local options⟩](⟨A,B⟩) ou (⟨A,B,C⟩)`

Attention les arguments sont des listes de deux ou bien de trois points. Les cercles que l'on peut tracer sont les mêmes que pour la macro précédente. Une option supplémentaire `R` afin de donner directement une mesure.

options	défaut	définition
through	through	cercle avec deux points définissant un rayon
diameter	through	cercle avec deux points définissant un diamètre
R	through	cercle caractérisé par un point et la mesure d'un rayon

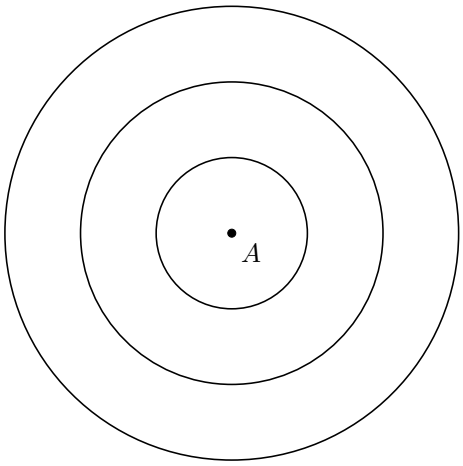
Il faut ajouter bien sûr tous les styles de Ti*k*Zpour les tracés

22.2.1 Cercles définis par un triangle.



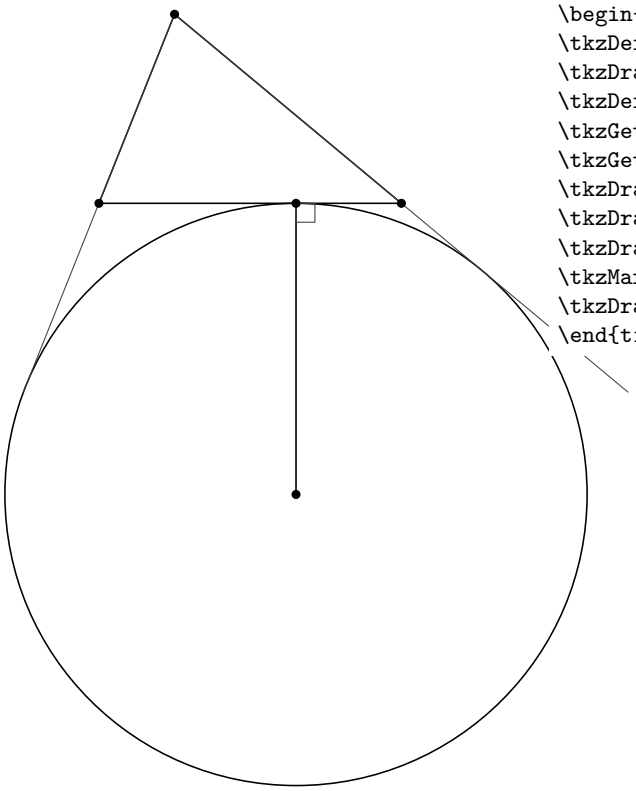

```
\begin{tikzpicture}
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(2,0){B}
  \tkzDefPoint(3,2){C}
  \tkzDrawPolygon(A,B,C)
  \tkzDrawCircles(A,B B,C C,A)
  \tkzDrawPoints(A,B,C)
  \tkzLabelPoints(A,B,C)
\end{tikzpicture}
```

22.2.2 Cercles concentriques.




```
\begin{tikzpicture}
   \tkzDefPoint(0,0){A}
   \tkzDrawCircles[R](A,1cm A,2cm A,3cm)
   \tkzDrawPoint(A)
   \tkzLabelPoints(A)
\end{tikzpicture}
```

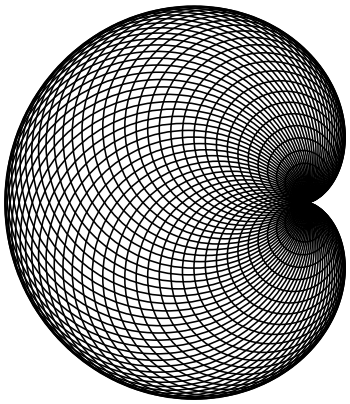
22.2.3 Cercles exinscrits.



```
\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{Jc} \tkzGetSecondPoint{Tc}
\tkzGetLength{rJc}
\tkzDrawCircle[R](Jc,{\rJc pt})
\tkzDrawLines[add=0 and 1](C,A C,B)
\tkzDrawSegment(Jc,Tc)
\tkzMarkRightAngle(Jc,Tc,B)
\tkzDrawPoints(A,B,C,Jc,Tc)
\end{tikzpicture}
```

22.2.4 Cardioïde

D'après une idée d'O. Reboux réalisée avec pst-eucl (module de Pstricks) de D. Rodriguez.
Son nom vient du grec kardia (cœur), en référence à sa forme, et lui fut donné par 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}
```

22.3 Tracer un demi-cercle

`\tkzDrawSemiCircle[⟨local options⟩](⟨A,B⟩) ou (⟨A,B,C⟩)`

Attention les arguments sont des listes de deux ou bien de trois points. Cette macro est, soit utilisée en partenariat avec **\tkzGetPoint** et/ou **\tkzGetLength** pour obtenir le centre et le rayon du cercle, soit en utilisant **tkzPointResult** et **tkzLengthResult** s'il n'est pas nécessaire de conserver les résultats.

options	défaut	définition
through	through	cercle caractérisé par deux points définissant un rayon
diameter	through	cercle caractérisé par deux points définissant un diamètre

22.4 Colorier un disque

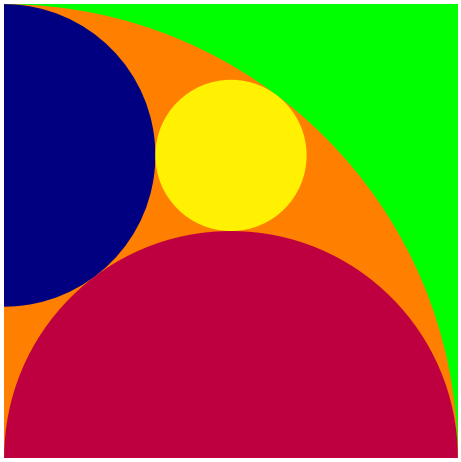
C'était possible avec la macro précédente, mais le tracé du disque était obligatoire, là ce n'est plus le cas.

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

options	défaut	définition
radius	radius	deux points définissent un rayon
R	radius	un point et la mesure d'un rayon

Il n'est pas nécessaire de mettre **radius** car c'est l'option par défaut. Il faut ajouter bien sûr tous les styles de Ti*k*Z pour les tracés

22.4.1 Exemple de \tkzFillCircle provenant d'un sangaku



```
\begin{tikzpicture}
   \tkzInit[xmin=0,xmax = 6,ymin=0,ymax=6]
   \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}
   \tkzDefTangent[from = B](F,A) \tkzGetPoints{G}{H}
   \tkzInterLL(F,G)(C,D) \tkzGetPoint{J}
   \tkzInterLL(A,J)(F,E) \tkzGetPoint{K}
   \tkzDefPointBy[projection=onto B--A](K)
 \tkzGetPoint{M}
   \tkzFillPolygon[color = green](A,B,C,D)
   \tkzFillCircle[color = orange](B,A)
   \tkzFillCircle[color = blue!50!black](M,A)
   \tkzFillCircle[color = purple](E,B)
   \tkzFillCircle[color = yellow](K,Q)
\end{tikzpicture}
```

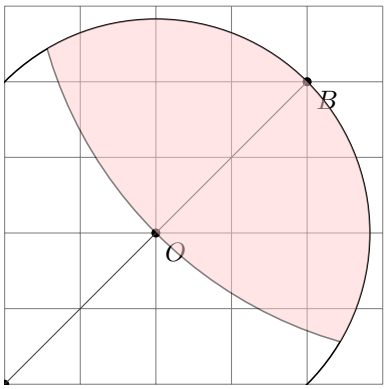
22.5 Clipper un disque

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

options	défaut	définition
radius	radius	cercle caractérisé par deux points définissant un rayon
R	radius	cercle caractérisé par un point et la mesure d'un rayon

Il n'est pas nécessaire de mettre **radius** car c'est l'option par défaut.

22.5.1 Exemple 1 de \tkzClipCircle




```
\begin{tikzpicture}
\tkzInit[xmax=5,ymax=5]
\tkzGrid \tkzClip
\tkzDefPoint(0,0){A}
\tkzDefPoint(2,2){O}
\tkzDefPoint(4,4){B}
\tkzDefPoint(6,6){C}
\tkzDrawPoints(O,A,B,C)
\tkzLabelPoints(O,A,B,C)
\tkzDrawCircle(O,A)
\tkzClipCircle(O,A)
\tkzDrawLine(A,C)
\tkzDrawCircle[fill=red!20,opacity=.5](C,O)
\end{tikzpicture}
```

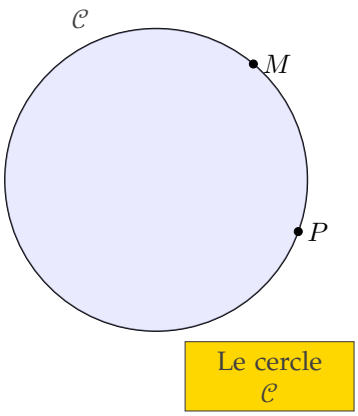
22.6 Donner un label à un cercle

`\tkzLabelCircle[⟨local options⟩](⟨A,B⟩)(⟨angle⟩){⟨label⟩}`

options	défaut	définition
radius	radius	cercle caractérisé par deux points définissant un rayon
R	radius	cercle caractérisé par un point et la mesure d'un rayon

Il n'est pas nécessaire de mettre **radius** car c'est l'option par défaut. On peut utiliser les styles de Ti*k*Z. Le label est créé et donc "passé" entre accolades.

22.6.1 Exemple de \tkzLabelCircle




```
\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!20,opacity=.4](O,M)
  \tkzLabelCircle[R,draw,fill=Gold,%
  text width=2cm,text centered](O,3 cm)(-60)%
          {Le cercle\\ $\mathcal{C}$}
  \tkzDrawPoints(M,P)\tkzLabelPoints[right](M,P)
\end{tikzpicture}
```

23 Intersections

Il est possible de déterminer les coordonnées des points d'intersection entre deux droites, une droite et un cercle et deux cercles.

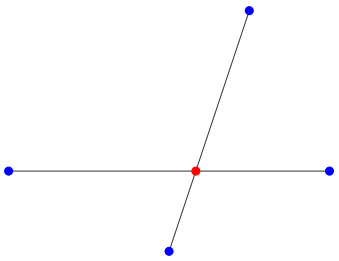
Les commandes associées n'ont pas d'arguments optionnels et l'usager doit lui même déterminer l'existence des points d'intersection.

23.1 Intersection de deux droites

\tkzInterLL(⟨A,B⟩)(⟨C,D⟩)

Définit le point d'intersection **tkzPointResult** des deux droites (AB) and (CD). Les points connus sont donnés en couple (deux par droite) entre parenthèses, quant au point obtenu, il peut être récupéré avec la macro **\tkzDefPoint**.

23.1.1 Exemple d'intersection entre deux droites



```
\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}
```

23.2 Intersection d'une droite et d'un cercle

Comme précédemment, la droite est définie par un couple de points. Le cercle est aussi défini par un un couple :

— (O,C) qui est un couple de points, le premier désigne le centre et le second est un point quelconque du cercle.

— (O,r) La mesure r est celle du rayon. Elle est exprimée soint en *cm*, soit en *pt*.

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

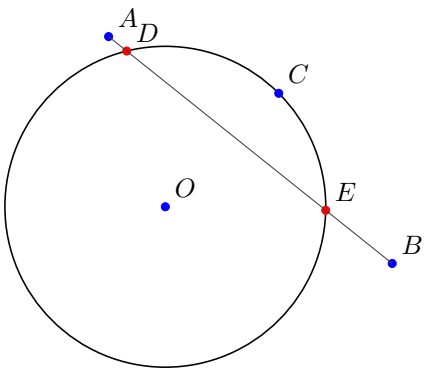
Les arguments sont donc deux couples.

options	défaut	définition
N	N	(O,C) détermine le cercle
R	N	(O, 1 cm) ou (O, 120 pt)
with nodes	N	(O,C,D) CD est un rayon

La macro définit les points d' intersection I et J de la droite (AB) et du cercle de centre O de rayon r s'ils existent ; dans le cas contraire, une erreur sera signalée dans le fichier .log

23.2.1 Exemple simple d'intersection droite-cercle

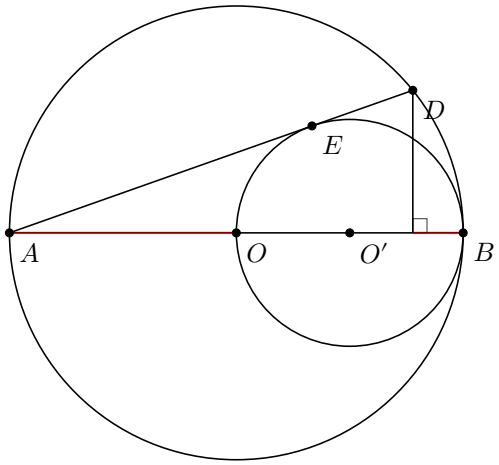
Dans l'exemple suivant, le tracé du cercle utilise deux points et l'intersection de la droite et du cercle utilise deux couples de points




```
\begin{tikzpicture}[scale=.75]
 \tkzInit[xmax=5,ymax=4]
 \tkzDefPoint(1,1){O}
 \tkzDefPoint(0,4){A}
 \tkzDefPoint(5,0){B}
 \tkzDefPoint(3,3){C}
 \tkzInterLC(A,B)(O,C)  \tkzGetPoints{D}{E}
 \tkzDrawCircle(O,C)
 \tkzDrawPoints[color=blue](O,A,B,C)
 \tkzDrawPoints[color=red](D,E)
 \tkzDrawLine(A,B)
 \tkzLabelPoints[above right](O,A,B,C,D,E)
\end{tikzpicture}
```

23.2.2 Exemple plus complexe d'intersection droite-cercle

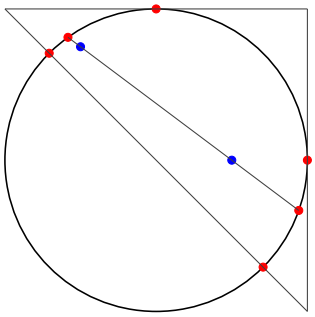
http://gogeometry.com/problem/p190_tangent_circle




```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(8,0){B}
  \tkzDefMidPoint(A,B)
  \tkzGetPoint{O}
  \tkzDrawCircle(O,B)
  \tkzDefMidPoint(O,B)
  \tkzGetPoint{O'}
  \tkzDrawCircle(O',B)
  \tkzDefTangent[from=A](O',B)
  \tkzGetSecondPoint{E}
  \tkzInterLC(A,E)(O,B)
  \tkzGetSecondPoint{D}
  \tkzDefPointBy[projection=onto A--B](D)
   \tkzGetPoint{F}
  \tkzMarkRightAngle(D,F,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)
  \tkzLabelPoints(A,B,O,O',E,D)
\end{tikzpicture}
```

23.2.3 Cercle défini par un centre et une mesure, et cas particuliers

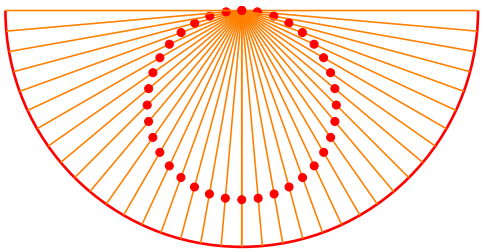
Regardons quelques cas particuliers comme des droites tangentes au cercle.



```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoint(0,8){A}  \tkzDefPoint(8,0){B}
  \tkzDefPoint(8,8){C}  \tkzDefPoint(4,4){I}
  \tkzDefPoint(2,7){E}  \tkzDefPoint(6,4){F}
  \tkzDrawCircle[R](I,4 cm)
  \tkzInterLC[R](A,C)(I,4 cm)  \tkzGetPoints{I1}{I2}
  \tkzInterLC[R](B,C)(I,4 cm)  \tkzGetPoints{J1}{J2}
  \tkzInterLC[R](A,B)(I,4 cm)  \tkzGetPoints{K1}{K2}
  \tkzDrawPoints[color=red](I1,J1,K1,K2)
  \tkzDrawLines(A,B B,C A,C)
  \tkzInterLC[R](E,F)(I,4 cm)  \tkzGetPoints{I2}{J2}
  \tkzDrawPoints[color=blue](E,F)
  \tkzDrawPoints[color=red](I2,J2)
  \tkzDrawLine(I2,J2)
\end{tikzpicture}
```

23.2.4 Exemple plus complexe

Attention à la syntaxe. Tout d'abord, les calculs pour les points peuvent être faits pendant le passage des arguments, mais il faut respecter la syntaxe de **xfp**. Vous pouvez constater que j'utilise le terme **pi** car travaille en radians !. De plus quand des calculs nécéssitent l'emploi de parenthèses, celles-ci doivent être insérées dans un groupe TEX{ ...}.



```
\begin{tikzpicture}[scale=1.25]
  \tkzDefPoint(0,1){J}
  \tkzDefPoint(0,0){O}
  \tkzDrawArc[R,line width=1pt,color=red](J,2.5 cm)(180,0)
  \foreach \i in {0,-5,-10,...,-85,-90}{
    \tkzDefPoint({2.5*cosd(\i)},{1+2.5*sind(\i)}){P}
     \tkzDrawSegment[color=orange](J,P)
     \tkzInterLC[R](P,J)(O,1 cm)
     \tkzGetPoints{M}{N}
     \tkzDrawPoints[red](N)
     }
  \foreach \i in {-90,-95,...,-175,-180}{
     \tkzDefPoint({2.5*cosd(\i)},{1+2.5*sind(\i)}){P}
     \tkzDrawSegment[color=orange](J,P)
     \tkzInterLC[R](P,J)(O,1 cm)
     \tkzGetPoints{M}{N}
     \tkzDrawPoints[red](M)
     }
\end{tikzpicture}
```

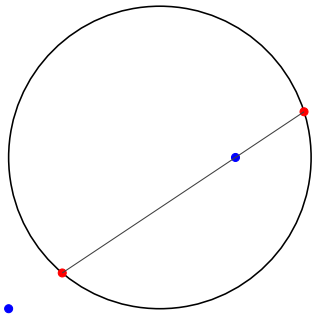
23.2.5 Calcul de la mesure du rayon

Avec **pgfmath** et **\pgfmathsetmacro**

La mesure du rayon peut être le résultat d'un calcul que l'on ne fera pas au sein de la macro d'intersection, mais avant. On peut calculer une longueur de plusieurs façons. Il est possible bien sûr, d'utiliser le module **pgfmath** et la macro **\pgfmathsetmacro**. Dans certains, les résultats obtenus ne sont pas assez précis ainsi le calcul suivant $0.0002 \div 0.0001$ donne 1.98 avec pgfmath alors que xfp donnera 2.

23.2.6 Calcul de la mesure du rayon

Avec **fp** et **\FPeval**

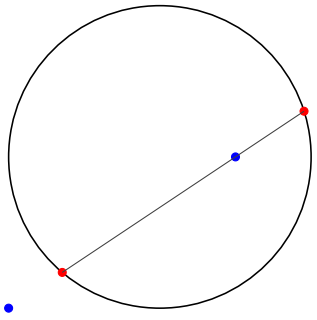


```
\begin{tikzpicture}
\tkzDefPoint(2,2){A}
\tkzDefPoint(5,4){B}
\tkzDefPoint(4,4){O}
\edef\tkzLen{\fpeval{0.0002/0.0001}}
\tkzDrawCircle[R](O,\tkzLen cm)
\tkzInterLC[R](A,B)(O, \tkzLen cm)
\tkzGetPoints{I}{J}
\tkzDrawPoints[color=blue](A,B)
\tkzDrawPoints[color=red](I,J)
\tkzDrawLine(I,J)
\end{tikzpicture}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

23.2.7 Calcul de la mesure du rayon

Avec TEX et **\tkzLength**.

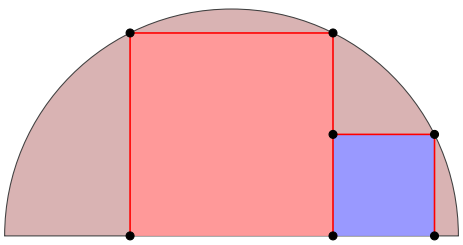
Cette dimension a été créée avec **\newdimen**. 2 cm a été transformé en points. Il est bien sûr possible d'utiliser TEX pour calculer.



```
\begin{tikzpicture}
\tkzDefPoints{2/2/A,5/4/B,4/4/O}
  \tkzLength=2cm
  \tkzDrawCircle[R](O,\tkzLength pt)
  \tkzInterLC[R](A,B)(O,\tkzLength pt)
  \tkzGetPoints{I}{J}
  \tkzDrawPoints[color=blue](A,B)
  \tkzDrawPoints[color=red](I,J)
  \tkzDrawLine(I,J)
\end{tikzpicture}
```

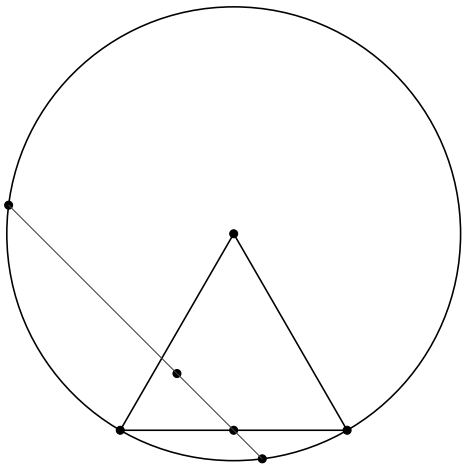
23.2.8 Des carrés dans un demi-disque

Un air de Sangaku ! Il s'agit de prouver que l'on peut inscrire dans un demi-disque, deux carrés, et de déterminer la longueur de leurs côtés respectifs en fonction du rayon.



```
\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[fill=Maroon!30](I,B)(A)
 \tkzFillPolygon[color=red!40](H,E,F,G)
 \tkzFillPolygon[color=blue!40](H,J,K,L)
 \tkzDrawPolySeg[color=red](H,E,F,G)
 \tkzDrawPolySeg[color=red](J,K,L)
 \tkzDrawPoints(E,G,H,F,J,K,L)
\end{tikzpicture}
```

23.2.9 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}
\tkzDrawCircle(C,A)
\tkzInterLC[with nodes](D,E)(C,A,B)
\tkzGetPoints{F}{G}
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints(A,...,G)
\tkzDrawLine(F,G)
\end{tikzpicture}
```

23.3 Intersection de deux cercles

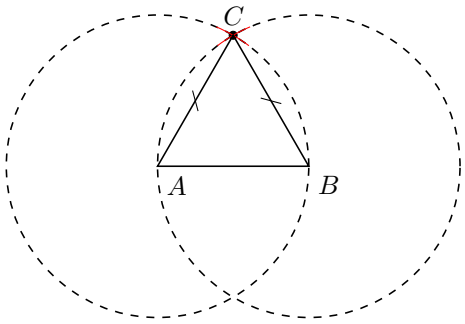
Le cas le plus fréquent est celui de deux cercles définis par leur centre et un point, mais comme précédemment l'option R permet d'utiliser les mesures des rayons

\tkzInterCC[⟨options⟩](⟨O, A/r⟩)(⟨O′, A′/r′⟩){⟨I⟩}{⟨J⟩}

options	défaut	définition
N	N	OA et O'A' sont des rayons, O et O' les centres
R	N	r et r' sont des dimensions et mesurent les rayons
with nodes	N	r et r' sont des dimensions et mesurent les rayons

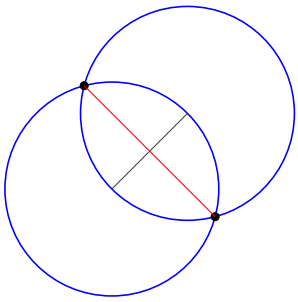
Cette macro définit le(s) point(s) d' intersection I et J des deux cercles de centre O et O'. Si les deux cercles n'ont pas de point commun alors la macro se termine par une erreur qui n'est pas gérée.
Il est également possible d'utiliser directement \tkzInterCCN et \tkzInterCCR.

23.3.1 Construction d'un triangle équilatéral



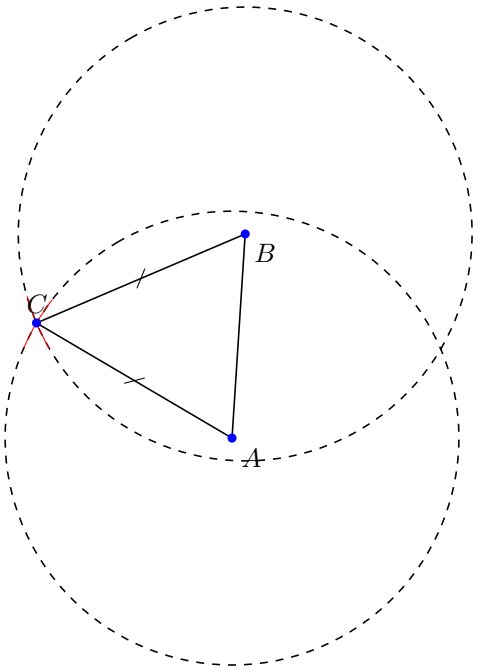

```
\begin{tikzpicture}[trim left=-1cm,scale=.5]
 \tkzDefPoint(1,1){A}
 \tkzDefPoint(5,1){B}
 \tkzInterCC(A,B)(B,A)\tkzGetPoints{C}{D}
 \tkzDrawPoint[color=black](C)
 \tkzDrawCircle[dashed](A,B)
 \tkzDrawCircle[dashed](B,A)
 \tkzCompass[color=red](A,C)
 \tkzCompass[color=red](B,C)
 \tkzDrawPolygon(A,B,C)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

23.3.2 Exemple une médiatrice



```
\begin{tikzpicture}[scale=.5]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(2,2){B}
  \tkzDrawCircle[color=blue](B,A)
  \tkzDrawCircle[color=blue](A,B)
  \tkzInterCC(B,A)(A,B)\tkzGetPoints{M}{N}
  \tkzDrawLine(A,B)
  \tkzDrawPoints(M,N)
  \tkzDrawLine[color=red](M,N)
\end{tikzpicture}
```

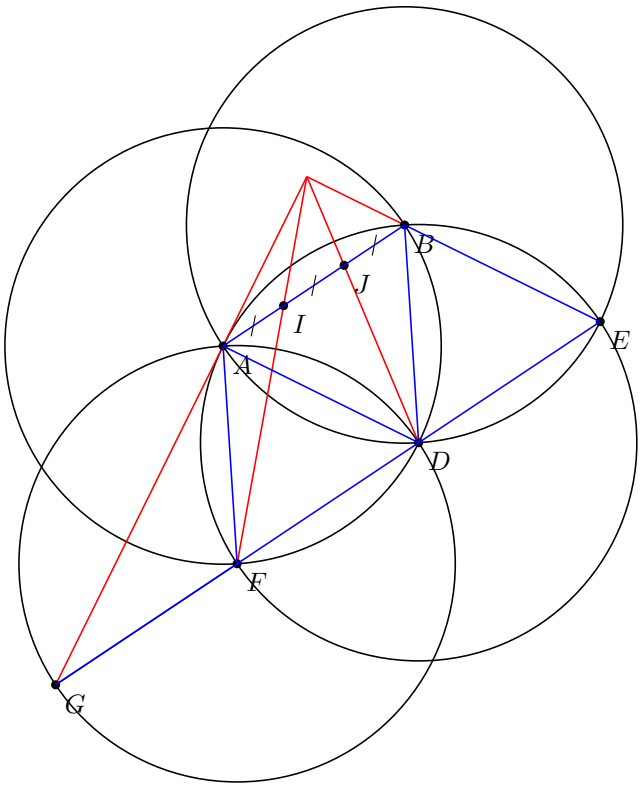
23.3.3 Un triangle isocèle.




```
\begin{tikzpicture}[rotate=120,scale=.75]
 \tkzDefPoint(1,2){A}
 \tkzDefPoint(4,0){B}
 \tkzInterCC[R](A,4cm)(B,4cm)
 \tkzGetPoints{C}{D}
 \tkzDrawCircle[R,dashed](A,4 cm)
 \tkzDrawCircle[R,dashed](B,4 cm)
 \tkzCompass[color=red](A,C)
 \tkzCompass[color=red](B,C)
 \tkzDrawPolygon(A,B,C)
 \tkzDrawPoints[color=blue](A,B,C)
 \tkzMarkSegments[mark=s|](A,C B,C)
 \tkzLabelPoints[](A,B)
 \tkzLabelPoint[above](C){$C$}
\end{tikzpicture}
```

23.3.4 Trisection d'un segment

Il s'agit ici de partager avec une règle et un compas, un segment en trois segments de même longueur.




```
\begin{tikzpicture}[scale=.8]
 \tkzDefPoint(0,0){A}
 \tkzDefPoint(3,2){B}
 \tkzInterCC(A,B)(B,A)
 \tkzGetPoints{C}{D}
 \tkzInterCC(D,B)(B,A)
 \tkzGetPoints{A}{E}
 \tkzInterCC(D,B)(A,B)
 \tkzGetPoints{F}{B}
 \tkzInterLC(E,F)(F,A)
 \tkzGetPoints{D}{G}
 \tkzInterLL(A,G)(B,E)
 \tkzGetPoint{O}
 \tkzInterLL(O,D)(A,B)
 \tkzGetPoint{J}
 \tkzInterLL(O,F)(A,B)
 \tkzGetPoint{I}
 \tkzDrawCircle(D,A)
 \tkzDrawCircle(A,B)
 \tkzDrawCircle(B,A)
 \tkzDrawCircle(F,A)
 \tkzDrawSegments[color=red](O,G
  O,B O,D O,F)
 \tkzDrawPoints(A,B,D,E,F,G,I,J)
 \tkzLabelPoints(A,B,D,E,F,G,I,J)
 \tkzDrawSegments[blue](A,B B,D A,D%
  A,F F,G E,G B,E)
 \tkzMarkSegments[mark=s|](A,I I,J J,B)
\end{tikzpicture}
```

24 Les angles

24.1 Colorier un angle : fill

L'opération la plus simple

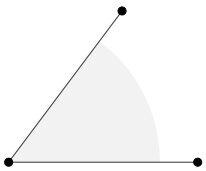
`\tkzFillAngle[⟨local options⟩](⟨A,O,B⟩)`

O est le sommet de l'angle. OA et OB sont les côtés. Attention l'angle est déterminé avec l'ordre des points.

options	défaut	définition
size	1 cm	cette option détermine le rayon du secteur angulaire colorié

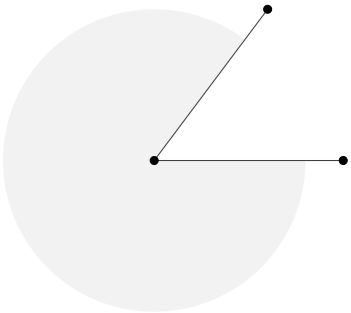
Il faut ajouter bien sûr tous les styles de Ti*k*Z comme par exemple l'usage de fill ou encore shade

24.1.1 Exemple avec size

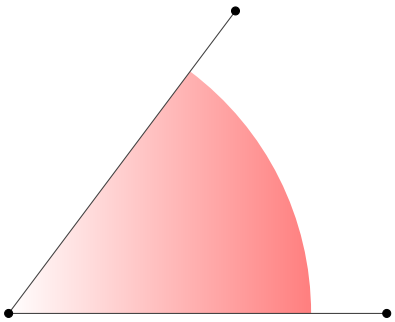


```
\begin{tikzpicture}
   \tkzInit
   \tkzDefPoints{0/0/O,2.5/0/A,1.5/2/B}
   \tkzFillAngle[size=2cm, fill=gray!10](A,O,B)
   \tkzDrawLines(O,A O,B)
   \tkzDrawPoints(O,A,B)
\end{tikzpicture}
```

24.1.2 Changement de l'ordre des points



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

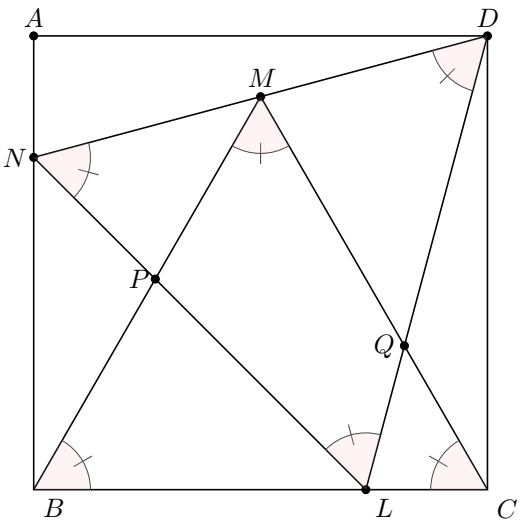


```
\begin{tikzpicture}
   \tkzInit
   \tkzDefPoints{0/0/O,5/0/A,3/4/B}
   % Don't forget {} to get, () to use
   \tkzFillAngle[size=4cm,left color=white,
                 right color=red!50](A,O,B)
   \tkzDrawLines(O,A O,B)
   \tkzDrawPoints(O,A,B)
\end{tikzpicture}
```

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

Avec des options communes, il existe une macro pour de mutiples angles

24.1.3 Multiples angles




```
\begin{tikzpicture}[scale=0.75]
  \tkzDefPoint(0,0){B}
  \tkzDefPoint(8,0){C}
  \tkzDefPoint(0,8){A}
  \tkzDefPoint(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}
```

24.2 Marquer un angle mark

Opération plus délicate car les options sont nombreuses. Les symboles utilisés pour le marquage outre ceux de TikZ sont définis dans le fichier `tkz-lib-marks.tex` et désignés par les caractères suivants :

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

Leurs définitions est la suivante

```
\pgfdeclareplotmark{||}
  %double bar
{%
  \pgfpathmoveto{\pgfqpoint{2\pgflinewidth}{\pgfplotmarksize}}
  \pgfpathlineto{\pgfqpoint{2\pgflinewidth}{-\pgfplotmarksize}}
  \pgfpathmoveto{\pgfqpoint{-2\pgflinewidth}{\pgfplotmarksize}}
  \pgfpathlineto{\pgfqpoint{-2\pgflinewidth}{-\pgfplotmarksize}}
  \pgfusepathqstroke
}


  %triple bar
  \pgfdeclareplotmark{|||}
  {%
    \pgfpathmoveto{\pgfqpoint{0 pt}{\pgfplotmarksize}}
    \pgfpathlineto{\pgfqpoint{0 pt}{-\pgfplotmarksize}}
    \pgfpathmoveto{\pgfqpoint{-3\pgflinewidth}{\pgfplotmarksize}}
    \pgfpathlineto{\pgfqpoint{-3\pgflinewidth}{-\pgfplotmarksize}}
    \pgfpathmoveto{\pgfqpoint{3\pgflinewidth}{\pgfplotmarksize}}
    \pgfpathlineto{\pgfqpoint{3\pgflinewidth}{-\pgfplotmarksize}}
    \pgfusepathqstroke
  }


  % An bar slant
  \pgfdeclareplotmark{s|}
  {%
    \pgfpathmoveto{\pgfqpoint{-.70710678\pgfplotmarksize}%
                             {-.70710678\pgfplotmarksize}}
    \pgfpathlineto{\pgfqpoint{.70710678\pgfplotmarksize}%
                             {.70710678\pgfplotmarksize}}
    \pgfusepathqstroke
  }


  % An double bar slant
  \pgfdeclareplotmark{s||}
  {%
   \pgfpathmoveto{\pgfqpoint{-0.75\pgfplotmarksize}{-\pgfplotmarksize}}
   \pgfpathlineto{\pgfqpoint{0.25\pgfplotmarksize}{\pgfplotmarksize}}
   \pgfpathmoveto{\pgfqpoint{0\pgfplotmarksize}{-\pgfplotmarksize}}
   \pgfpathlineto{\pgfqpoint{1\pgfplotmarksize}{\pgfplotmarksize}}
   \pgfusepathqstroke
  }
```

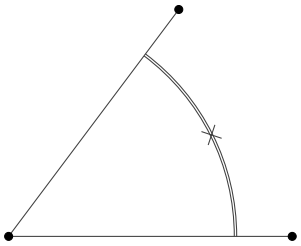
```
% z
\pgfdeclareplotmark{z}
{%
  \pgfpathmoveto{\pgfqpoint{0.75\pgfplotmarksize}{-\pgfplotmarksize}}
  \pgfpathlineto{\pgfqpoint{-0.75\pgfplotmarksize}{-\pgfplotmarksize}}
  \pgfpathlineto{\pgfqpoint{0.75\pgfplotmarksize}{\pgfplotmarksize}}
  \pgfpathlineto{\pgfqpoint{-0.75\pgfplotmarksize}{\pgfplotmarksize}}
  \pgfusepathqstroke
}



% s
\pgfdeclareplotmark{s}
{%
   \pgfpathmoveto{\pgfqpoint{0pt}{0pt}}
   \pgfpathcurveto
       {\pgfpoint{0pt}{0pt}}
       {\pgfpoint{-\pgfplotmarksize}{\pgfplotmarksize}}
       {\pgfpoint{\pgfplotmarksize}{\pgfplotmarksize}}
   \pgfpathmoveto{\pgfqpoint{0pt}{0pt}}
    \pgfpathcurveto
       {\pgfpoint{0pt}{0pt}}
       {\pgfpoint{\pgfplotmarksize}{-\pgfplotmarksize}}
       {\pgfpoint{-\pgfplotmarksize}{-\pgfplotmarksize}}
    \pgfusepathqstroke
}



% infinity
\pgfdeclareplotmark{oo}
{%
   \pgfpathmoveto{\pgfqpoint{0pt}{0pt}}
   \pgfpathcurveto
       {\pgfpoint{0pt}{0pt}}
       {\pgfpoint{.5\pgfplotmarksize}{1\pgfplotmarksize}}
       {\pgfpoint{\pgfplotmarksize}{0pt}}
   \pgfpathmoveto{\pgfqpoint{0pt}{0pt}}
    \pgfpathcurveto
       {\pgfpoint{0pt}{0pt}}
       {\pgfpoint{-.5\pgfplotmarksize}{1\pgfplotmarksize}}
       {\pgfpoint{-\pgfplotmarksize}{0pt}}
   \pgfpathmoveto{\pgfqpoint{0pt}{0pt}}
      \pgfpathcurveto
       {\pgfpoint{0pt}{0pt}}
       {\pgfpoint{.5\pgfplotmarksize}{-1\pgfplotmarksize}}
       {\pgfpoint{\pgfplotmarksize}{0pt}}
   \pgfpathmoveto{\pgfqpoint{0pt}{0pt}}
    \pgfpathcurveto
       {\pgfpoint{0pt}{0pt}}
       {\pgfpoint{-.5\pgfplotmarksize}{-1\pgfplotmarksize}}
       {\pgfpoint{-\pgfplotmarksize}{0pt}}
    \pgfusepathqstroke
}
```

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

O est le sommet. Attention les arguments varient en fonction des options. Plusieurs marquages sont possibles. Vous pouvez simplement tracer un arc ou bian ajouter une marque sur cet arc. Le style de l'arc est choisi avec l'option **arc**, le rayon de l'arc est donné par **mksize**, l'arc peut bien sûr être colorié.

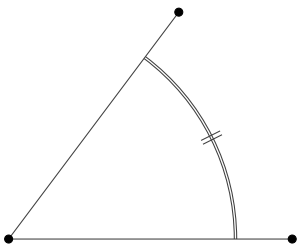
options	défaut	définition
arc	l	choix parmi l, ll et lll simple, double ou triple.
size	1 cm	rayon de l'arc.
mark	none	choix parmi s.
mksize	4pt	taille du symbol (mark).
mkcolor	black	couleur du symbole (mark).
mkpos	0.5	position du symbole sur l'arc.

24.2.1 Exemple avec mark = x



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

24.2.2 Exemple avec mark =||



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

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

Avec des options communes, il existe une macro pour de mutiples angles

24.3 Label dans un angle

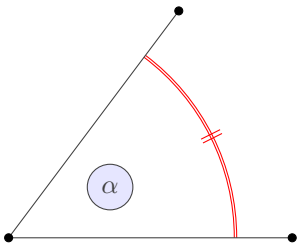
`\tkzLabelAngle[⟨local options⟩](⟨A,O,B⟩)`

Une seule option **dist** qui n'est pas indispensable car l'option **pos** de Ti*k*Z fonctionne très bien.

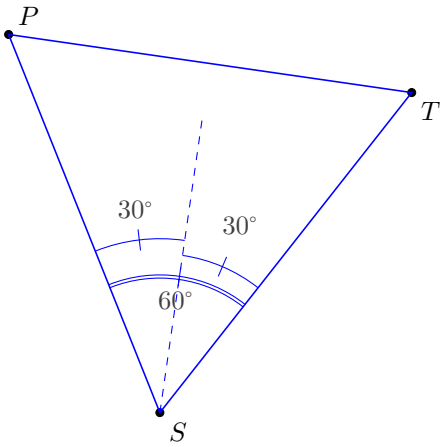
options	défaut	définition
pos	1	ou dist, permet de contrôler la distance du sommet au label.

Il est possible de déplacer le label avec toutes les options de TikZ : rotate, shift, below, etc.

24.3.1 Exemple avec pos




```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoints{0/0/O,5/0/A,3/4/B}
  \tkzMarkAngle[size = 4cm,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}
```




```
\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 = 1.8cm,mark = |,arc=ll,
                    color = blue](T,S,P)
  \tkzMarkAngle[size = 2.1cm,mark = |,arc=l,
                    color = blue](T,S,s)
  \tkzMarkAngle[size = 2.3cm,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.

Avec des options communes, il existe une macro pour de mutiples angles

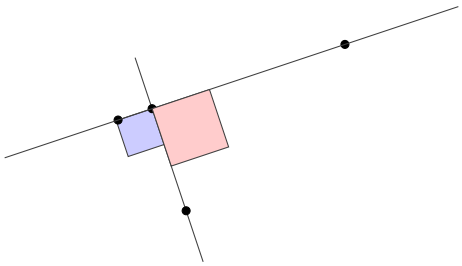
24.4 Marquer un angle droit

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

L'option **german** permet de changer le style du dessin. L'option **size** permet de modifier la taille du dessin.

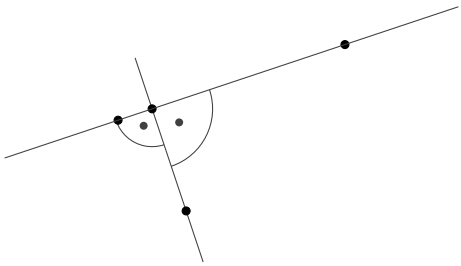
options	défaut	définition
german	normal	german arc avec point intérieur.
size	0.2	taille d'un coté.

24.4.1 Exemple de marquage d'un angle droit



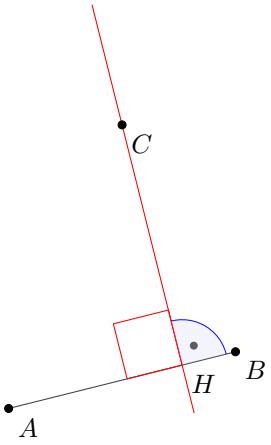
```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,3/1/B,0.9/-1.2/P}
  \tkzDefPointBy[projection = onto B--A](P)  \tkzGetPoint{H}
  \tkzDrawLines[add=.5 and .5](P,H)
  \tkzMarkRightAngle[fill=blue!20,size=.5,draw](A,H,P)
  \tkzDrawPoints[](A,B,P,H)
  \tkzDrawLines[add=.5 and .5](A,B)
  \tkzMarkRightAngle[fill=red!20,size=.8](B,H,P)
\end{tikzpicture}
```

24.4.2 Exemple de marquage d'un angle droit, german style



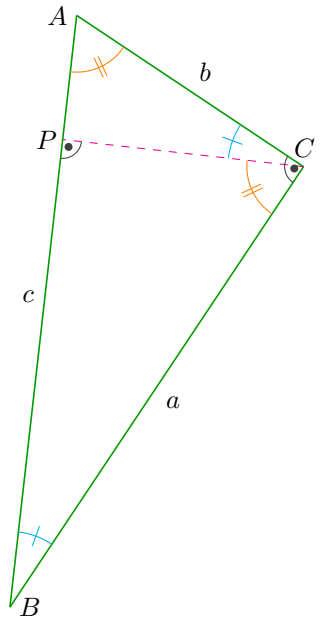
```
\begin{tikzpicture}
  \tkzDefPoints{0/0/A,3/1/B,0.9/-1.2/P}
  \tkzDefPointBy[projection = onto B--A](P)  \tkzGetPoint{H}
  \pgfresetboundingbox
  \tkzDrawLines[add=.5 and .5](P,H)
  \tkzMarkRightAngle[german,size=.5,draw](A,H,P)
  \tkzDrawPoints[](A,B,P,H)
  \tkzDrawLines[add=.5 and .5,fill=blue!20](A,B)
  \tkzMarkRightAngle[german,size=.8](P,H,B)
\end{tikzpicture}
```

24.4.3 Mélange de 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)
\end{tikzpicture}
```

24.4.4 Exemple complet




```
\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)
\tkzDrawLine[altitude,dashed,color=magenta](B,C,A)
\tkzGetPoint{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=1cm,color=cyan,mark=|](C,B,A)
\tkzMarkAngle[size=1cm,color=cyan,mark=|](A,C,P)
\tkzMarkAngle[size=0.75cm,color=orange,mark=||](P,C,B)
\tkzMarkAngle[size=0.75cm,color=orange,mark=||](B,A,C)
\tkzMarkRightAngle[german](A,C,B)
\tkzMarkRightAngle[german](B,P,C)
\end{tikzpicture}
```

24.5 \tkzMarkRightAngles

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

Avec des options communes, il existe une macro pour de mutiples angles

24.6 \tkzGetAngle

\tkzGetAngle(⟨macro⟩)

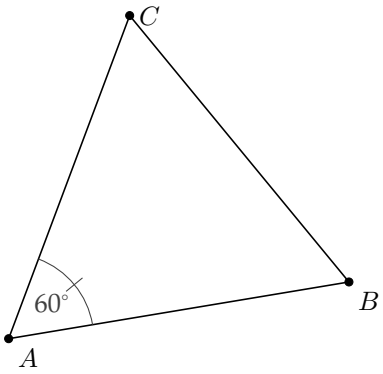
Attribue la valeur d'un angle à une macro.

24.7 \tkzFindAngle

\tkzFindAngle(⟨A,O,B⟩)

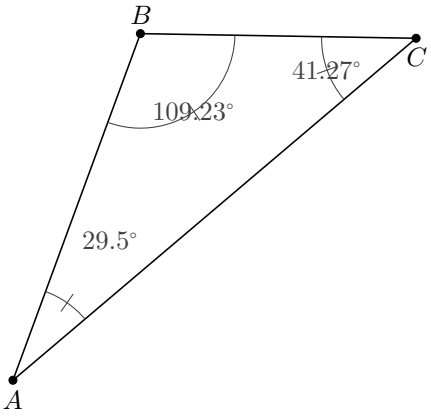
Détermine la valeur de l'angle en degrés.

24.7.1 Vérication de la mesure d'un angle




```
\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=1.5cm](B,A,C)
\end{tikzpicture}
```

24.7.2 Détermination des trois angles d'un triangle



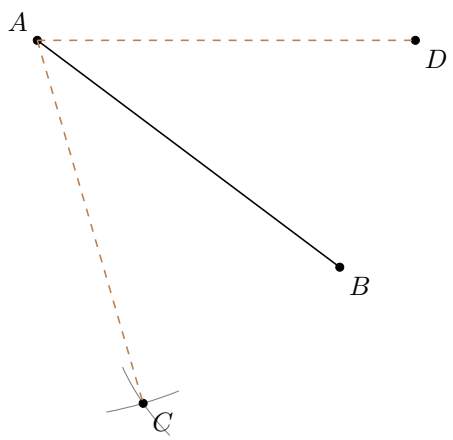

```
\begin{tikzpicture}[scale=1.25,rotate=30]
\tkzDefPoints{0.5/1.5/A, 3.5/4/B, 6/2.5/C}
\tkzDrawPolygon(A,B,C)
\tkzDrawPoints(A,B,C)
\tkzLabelPoints[below](A,C)
\tkzLabelPoints[above](B)
\tkzMarkAngle[size=1cm](B,C,A)
\tkzFindAngle(B,C,A)
\tkzGetAngle{angleBCA}
\edef\angleBCA{\fpeval{round(\angleBCA,2)}}
\tkzLabelAngle[pos = 1](B,C,A){$\angleBCA^{\circ}$}
\tkzMarkAngle[size=1cm](C,A,B)
\tkzFindAngle(C,A,B)
\tkzGetAngle{angleBAC}
\edef\angleBAC{\fpeval{round(\angleBAC,2)}}
\tkzLabelAngle[pos = 1.8](C,A,B){%
          $\angleBAC^{\circ}$}
\tkzMarkAngle[size=1cm](A,B,C)
\tkzFindAngle(A,B,C)
\tkzGetAngle{angleABC}
\edef\angleABC{\fpeval{round(\angleABC,2)}}
\tkzLabelAngle[pos = 1](A,B,C){$\angleABC^{\circ}$}
\end{tikzpicture}
```

24.8 \tkzFindSlopeAngle

\tkzFindSlopeAngle(⟨A,B⟩)

Détermine la pente de la droite (AB).

24.8.1 Pliage




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

25 Les secteurs

25.1 \tkzDrawSector

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

Attention les arguments varient en fonction des options.

options	défaut	définition
towards	towards	O est le centre et l'arc par de A vers (OB)
rotate	towards	l'arc part de A et l'angle détermine sa longueur
R	towards	On donne le rayon et deux angles
R with nodes	towards	On donne le rayon et deux points

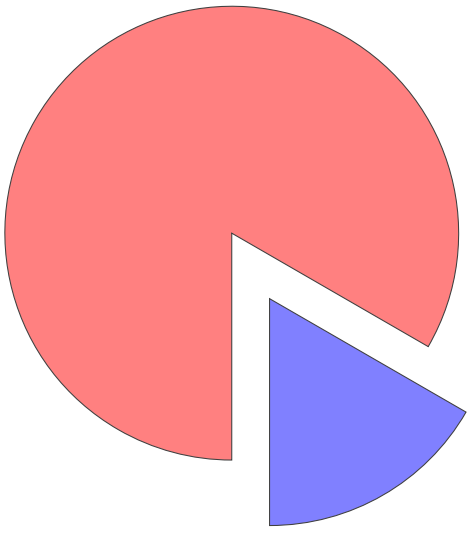
Il faut ajouter bien sûr tous les styles de Ti*k*Z pour les tracés

options	arguments	exemple
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=blue](O,2 cm)(30,90)
R with nodes	(⟨pt,r⟩)(⟨pt,pt⟩)	\tkzDrawSector[R with nodes](O,2 cm)(A,B)

Quelques exemples :

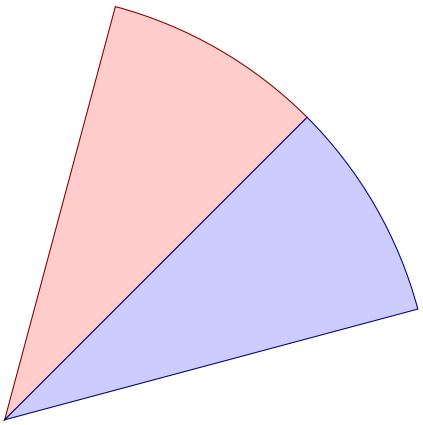
25.1.1 \tkzDrawSector et towards

Il est inutile de mettre **towards**. Il est possible d'utiliser **fill** en option.



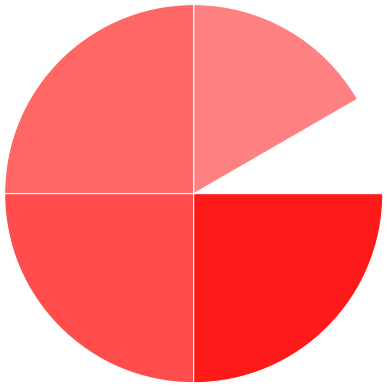
```
\begin{tikzpicture}[scale=1]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(-30:3){A}
  \tkzDefPointBy[rotation = center O angle -60](A)
  \tkzDrawSector[fill=red!50](O,A)(tkzPointResult)
 \begin{scope}[shift={(-60:1cm)}]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(-30:3){A}
  \tkzDefPointBy[rotation = center O angle -60](A)
  \tkzDrawSector[fill=blue!50](O,tkzPointResult)(A)
  \end{scope}
\end{tikzpicture}
```

25.1.2 \tkzDrawSector et rotate



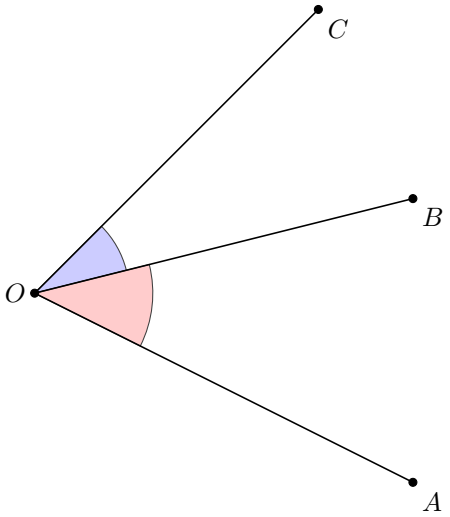
```
\begin{tikzpicture}[scale=2]
 \tkzDefPoint(0,0){O}
 \tkzDefPoint(2,2){A}
 \tkzDrawSector[rotate,draw=red!50!black,%
 fill=red!20](O,A)(30)
 \tkzDrawSector[rotate,draw=blue!50!black,%
 fill=blue!20](O,A)(-30)
\end{tikzpicture}
```

25.1.3 \tkzDrawSector et R



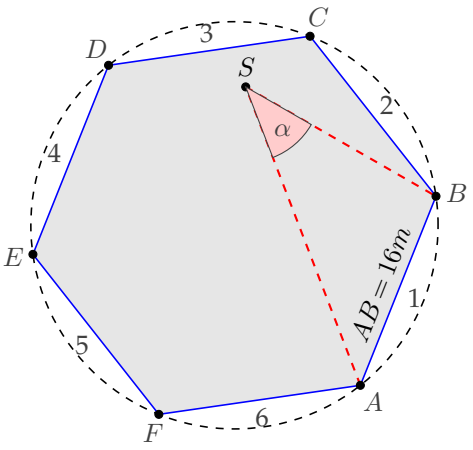
```
\begin{tikzpicture}[scale=1.25]
 \tkzDefPoint(0,0){O}
 \tkzDefPoint(2,-1){A}
 \tkzDrawSector[R,draw=white,%
 fill=red!50](O,2cm)(30,90)
 \tkzDrawSector[R,draw=white,%
 fill=red!60](O,2cm)(90,180)
 \tkzDrawSector[R,draw=white,%
 fill=red!70](O,2cm)(180,270)
 \tkzDrawSector[R,draw=white,%
 fill=red!90](O,2cm)(270,360)
\end{tikzpicture}
```

25.1.4 \tkzDrawSector et R




```
\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=blue!20](O,1 cm)(B,C)
 \tkzDrawSector[R with nodes,%
                fill=red!20](O,1.25 cm)(A,B)
\tkzDrawSegments(O,A O,B O,C)
\tkzDrawPoints(O,A,B,C)
\tkzLabelPoints(A,B,C)
\tkzLabelPoints[left](O)
\end{tikzpicture}
```

25.1.5 \tkzDrawSector et R with nodes




```
\begin{tikzpicture} [scale=.5]
 \tkzDefPoint(-1,-2){A}
 \tkzDefPoint(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 cm)(A,B)
  \tkzLabelAngle[pos=1.5](A,S,B){$\alpha$}
\end{tikzpicture}
```

25.2 \tkzFillSector

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

Attention les arguments varient en fonction des options.

options	défaut	définition
towards	towards	O est le centre et l'arc par de A vers (OB)
rotate	towards	l'arc part de A et l'angle détermine sa longueur
R	towards	On donne le rayon et deux angles
R with nodes	towards	On donne le rayon et deux points

Il faut ajouter bien sûr tous les styles de TikZpour les tracés

options	arguments	exemple
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 cm)(30,90)`
R with nodes	(⟨pt,r⟩)(⟨pt,pt⟩)	`\tkzFillSector[R with nodes](O,2 cm)(A,B)`

25.2.1 \tkzFillSector et towards

Il est inutile de mettre **towards** et vous remarquerez que les contours ne sont pas tracés,seule la surface est colorée.

```
\begin{tikzpicture}[scale=.6]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(-30:3){A}
  \tkzDefPointBy[rotation = center O angle -60](A)
  \tkzFillSector[fill=red!50](O,A)(tkzPointResult)
  \begin{scope}[shift={(-60:1cm)}]
   \tkzDefPoint(0,0){O}
   \tkzDefPoint(-30:3){A}
   \tkzDefPointBy[rotation = center O angle -60](A)
   \tkzFillSector[color=blue!50](O,tkzPointResult)(A)
  \end{scope}
\end{tikzpicture}
```

25.2.2 \tkzFillSector et rotate


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

25.3 \tkzClipSector

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

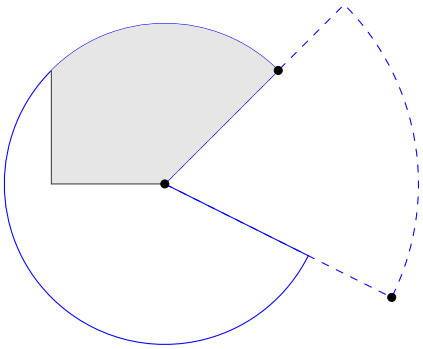
Attention les arguments varient en fonction des options.

options	défaut	définition
towards	towards	O est le centre et le secteur part de A vers (OB)
rotate	towards	le secteur part de A et l'angle détermine son amplitude
R	towards	On donne le rayon et deux angles

Il faut ajouter bien sûr tous les styles de Ti*k*Z pour les tracés

options	arguments	exemple
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 cm)(30,90)

25.3.1 \tkzClipSector



```
\begin{tikzpicture}[scale=1.5]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2,-1){A}
  \tkzDefPoint(1,1){B}
  \tkzDrawSector[color=blue,dashed](O,A)(B)
  \tkzDrawSector[color=blue](O,B)(A)
  \tkzClipBB
  \begin{scope}
    \tkzClipSector(O,B)(A)
    \draw[fill=gray!20] (-1,0) rectangle (3,3);
  \end{scope}
  \tkzDrawPoints(A,B,O)
\end{tikzpicture}
```

26 Les arcs

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

Cette macro trace un arc de centre O. Suivant les options, les arguments diffèrent. Il s'agit de déterminer un point de départ et un point d'arrivée. Soit le point de départ est donné, c'est ce qu'il y a de plus simple, soit on donne le rayon de l'arc. Dans ce dernier cas, il est nécessaire d'avoir deux angles. On peut soit donner directement les angles, soit donner des nodes qui associés au centre permettront de les déterminer.

options	défaut	définition
towards	towards	O est le centre et l'arc par de A vers (OB)
rotate	towards	l'arc part de A et l'angle détermine sa longueur
R	towards	On donne le rayon et deux angles
R with nodes	towards	On donne le rayon et deux points
delta	0	angle ajouté de chaque côté

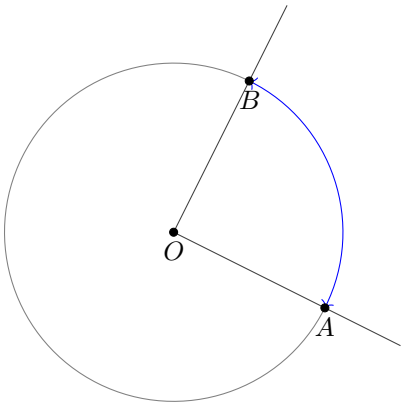
Il faut ajouter bien sûr tous les styles de Ti*k*Zpour les tracés

options	arguments	exemple
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,color=blue](O,2 cm)(30,90)
R with nodes	(⟨pt,r⟩)(⟨pt,pt⟩)	\tkzDrawArc[R with nodes](O,2 cm)(A,B)

Quelques exemples :

26.1 \tkzDrawArc et towards

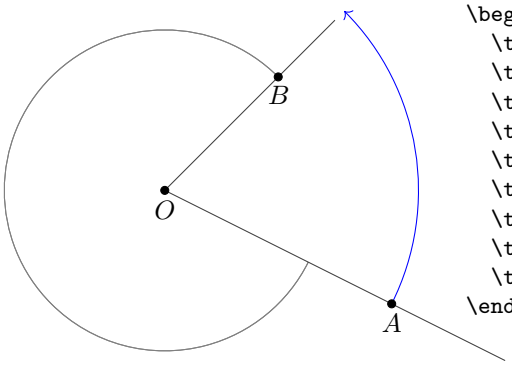
Il est inutile de mettre **towards**. Dans ce premier exemple l'arc part de A et va sur B. L'arc qui va de B vers A est différent. On obtient le saillant en allant dans le sens direct du cercle trigonométrique.




```
\begin{tikzpicture}
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2,-1){A}
  \tkzDefPointBy[rotation= center O angle 90](A)
  \tkzGetPoint{B}
  \tkzDrawArc[color=blue,<->](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}
```

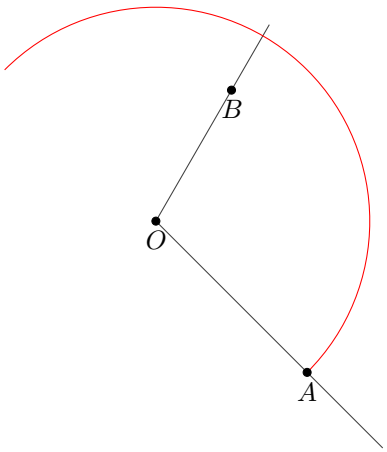
26.2 \tkzDrawArc et towards

Dans celui-ci, l'arc part de A mais s'arrête sur la droite (OB).



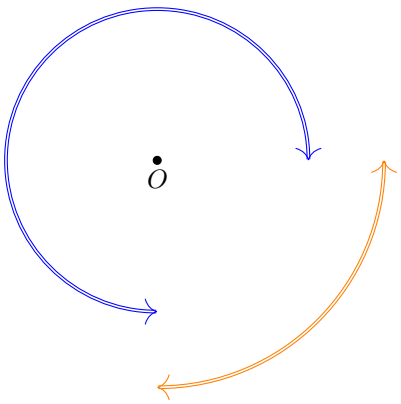

```
\begin{tikzpicture}[scale=1.5]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2,-1){A}
  \tkzDefPoint(1,1){B}
  \tkzDrawArc[color=blue,->](O,A)(B)
  \tkzDrawArc[color=gray](O,B)(A)
  \tkzDrawArc(O,B)(A)
  \tkzDrawLines[add = 0 and .5](O,A O,B)
  \tkzDrawPoints(O,A,B)
  \tkzLabelPoints[below](O,A,B)
\end{tikzpicture}
```

26.3 \tkzDrawArc et rotate



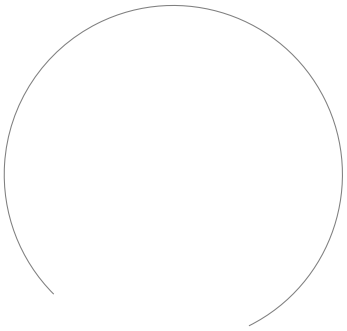

```
\begin{tikzpicture}
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2,-2){A}
  \tkzDefPoint(60:2){B}
  \tkzDrawLines[add = 0 and .5](O,A O,B)
  \tkzDrawArc[rotate,color=red](O,A)(180)
  \tkzDrawPoints(O,A,B)
  \tkzLabelPoints[below](O,A,B)
\end{tikzpicture}
```

26.4 \tkzDrawArc et R




```
\begin{tikzpicture}
  \tkzDefPoints{0/0/O}
  \tikzset{compass style/.append style={<->}}
  \tkzDrawArc[R,color=orange,double](O,3cm)(270,360)
  \tkzDrawArc[R,color=blue,double](O,2cm)(0,270)
  \tkzDrawPoint(O)
  \tkzLabelPoint[below](O){$O$}
\end{tikzpicture}
```

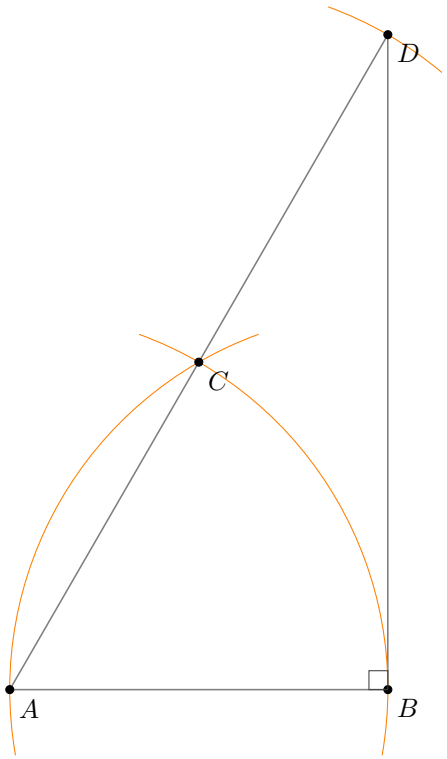
26.5 \tkzDrawArc et R with nodes



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

26.6 \tkzDrawArc et delta

Cette option permet un peu comme **\tkzCompass** de placer un arc et de déborder de chaque côté. delta est une mesure en degré.




```
\begin{tikzpicture}
 \tkzInit
 \tkzDefPoint(0,0){A}
 \tkzDefPoint(5,0){B}
 \tkzDefPointBy[rotation= center A angle 60](B)
 \tkzGetPoint{C}
 \tkzSetUpLine[color=gray]
 \tkzDefPointBy[symmetry= center C](A)
 \tkzGetPoint{D}
 \tkzDrawSegments(A,B A,D)
 \tkzDrawLine(B,D)
 \tkzSetUpCompass[color=orange]
 \tkzDrawArc[delta=10](A,B)(C)
 \tkzDrawArc[delta=10](B,C)(A)
 \tkzDrawArc[delta=10](C,D)(D)
 \tkzDrawPoints(A,B,C,D)
 \tkzLabelPoints(A,B,C,D)
 \tkzMarkRightAngle(D,B,A)
\end{tikzpicture}
```

27 Utilisation du compas

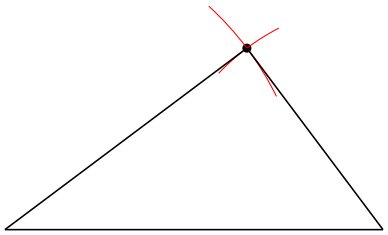
27.1 Macro principale \tkzCompass

\tkzCompass[⟨local options⟩](⟨A,B⟩)

Cette macro permet de laisser une trace de compas autrement dit un arc en un point désigné. Il faut indiquer le centre. Plusieurs options spécifiques vont modifier l'aspect de l'arc ainsi que les options de TikZ comme le style, la couleur, l'épaisseur du trait etc.

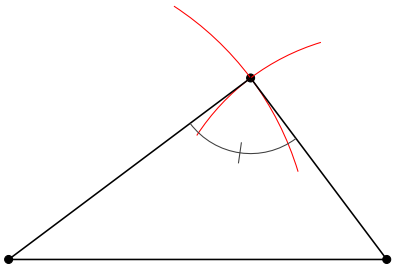
options	défaut	définition
delta	0	Modifie l'angle de l'arc en l'augmentant symétriquement
length	1	Modifie la longueur

27.1.1 Option length



```
\begin{tikzpicture}
  \tkzDefPoint(1,1){A}
  \tkzDefPoint(6,1){B}
  \tkzInterCC[R](A,4cm)(B,3cm)
  \tkzGetPoints{C}{D}
  \tkzDrawPoint(C)
  \tkzCompass[color=red,length=1.5](A,C)
  \tkzCompass[color=red](B,C)
  \tkzDrawSegments(A,B A,C B,C)
\end{tikzpicture}
```

27.1.2 Option delta



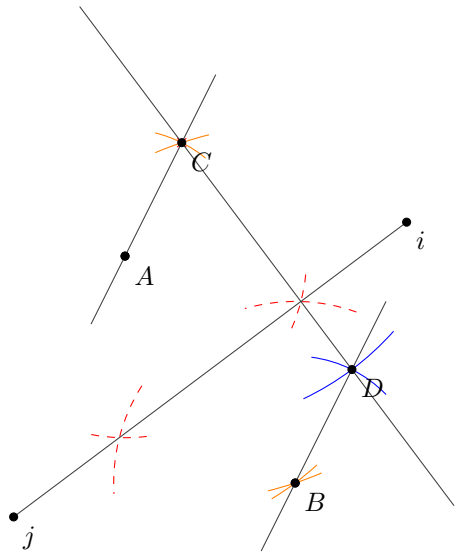
```
\begin{tikzpicture}
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(5,0){B}
  \tkzInterCC[R](A,4cm)(B,3cm)
  \tkzGetPoints{C}{D}
  \tkzDrawPoints(A,B,C)
  \tkzCompass[color=red,delta=20](A,C)
  \tkzCompass[color=red,delta=20](B,C)
  \tkzDrawPolygon(A,B,C)
  \tkzMarkAngle(A,C,B)
\end{tikzpicture}
```

27.2 Multiples constructions \tkzCompasss

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

Attention les arguments sont des listes de deux points. Cela permet d'économiser quelques lignes de codes.

options	défaut	définition
delta	0	Modifie l'angle de l'arc en l'augmentant symétriquement
length	1	Modifie la longueur



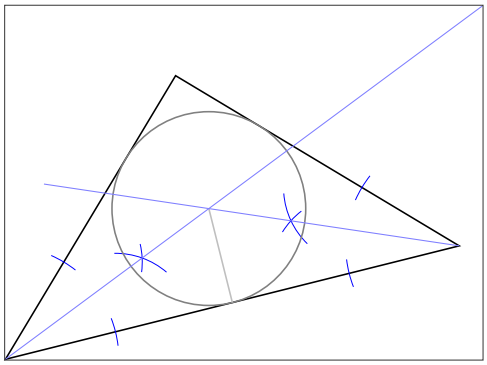

```
\begin{tikzpicture}[scale=.75]
 \tkzDefPoint(2,2){A}  \tkzDefPoint(5,-2){B}
 \tkzDefPoint(3,4){C}  \tkzDrawPoints(A,B)
 \tkzDrawPoint[color=red,shape=cross out](C)
 \tkzCompasss[color=orange](A,B A,C B,C C,B)
 \tkzShowLine[mediator,color=red,
                                dashed,length = 2](A,B)
 \tkzShowLine[parallel = through C,
                               color=blue,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}
```

27.3 Macro de configuration \tkzSetUpCompass

\tkzSetUpCompass[⟨local options⟩]

options	défaut	définition
line width	0.4pt	épaisseur du trait
color	black!50	couleur du trait
style	solid	style du trait solid, dashed,dotted,...

```
\tkzSetUpCompass[color=blue,line width=.3 pt]
```



```
\begin{tikzpicture}[scale=.75,
      showbi/.style={bisector,size=2,gap=3}]
  \tkzSetUpCompass[color=blue,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[radius,color=gray](I,H)
  \tkzDrawSegments[color=gray!50](I,H)
  \tkzDrawLines[add=0 and -.2,color=blue!50 ](A,a B,b)
\tkzShowBB
\end{tikzpicture}
```

28 The Show

28.1 Montrer les constructions de certaines lignes \tkzShowLine

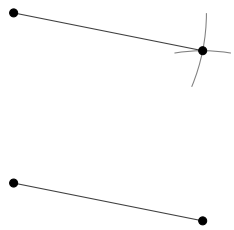
\tkzShowLine[⟨local options⟩](⟨pt1,pt2⟩) ou (⟨pt1,pt2,pt3⟩)

Ces constructions concernent les médiatrices, les droites perpendiculaires ou parallèles passant par un point donné et les bissectrices. Les arguments sont donc des listes de deux ou bien de trois points. Plusieurs options permettent l'ajustement des constructions. L'idée de cette macro revient à **Yves Combe**

options	défaut	définition
mediator	mediator	affiche les constructions d'une médiatrice
perpendicular	mediator	constructions pour une perpendiculaire
orthogonal	mediator	idem
bisector	mediator	constructions pour une bissectrice
K	1	cercle inscrit dans à un triangle
length	1	en cm, longueur d'un arc
ratio	.5	rapport entre les longueurs des arcs
gap	2	placement le point de construction
size	1	rayon d'un arc (voir bissectrice)

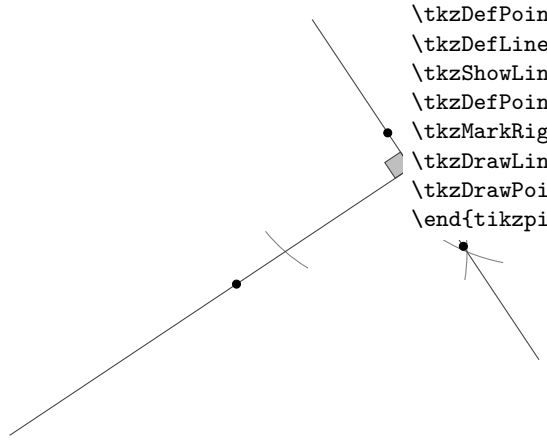
Il faut ajouter bien sûr tous les styles de Ti*k*Z pour les tracés

28.1.1 Exemple de \tkzShowLine et parallel



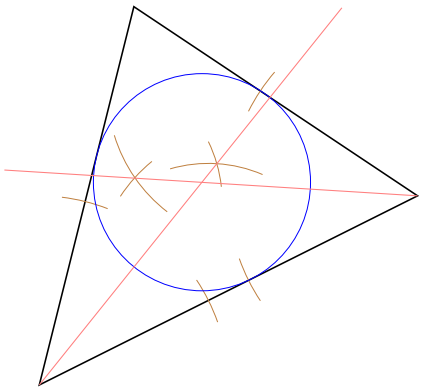
```
\begin{tikzpicture}
 \tkzDefPoints{-1.5/-0.25/A,1/-0.75/B,-1.5/2/C}
 \tkzDrawLine(A,B)
 \tkzDefLine[parallel=through C](A,B) \tkzGetPoint{c}
 \tkzShowLine[parallel=through C](A,B)
 \tkzDrawLine(C,c) \tkzDrawPoints(A,B,C,c)
\end{tikzpicture}
```

28.1.2 Exemple de \tkzShowLine et perpendicular



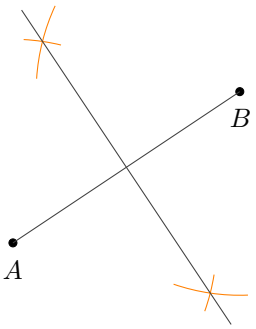
```
\begin{tikzpicture}
\tkzDefPoints{0/0/A, 3/2/B, 2/2/C}
\tkzDefLine[perpendicular=through C,K=-.5](A,B) \tkzGetPoint{c}
\tkzShowLine[perpendicular=through C,K=-.5,gap=3](A,B)
\tkzDefPointBy[projection=onto A--B](c)\tkzGetPoint{h}
\tkzMarkRightAngle[fill=lightgray](A,h,C)
\tkzDrawLines[add=1 and 1](A,B C,c)
\tkzDrawPoints(A,B,C,h,c)
\end{tikzpicture}
```

28.1.3 Exemple de \tkzShowLine et 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[radius,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}
```

28.1.4 Exemple de \tkzShowLine et 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}
```

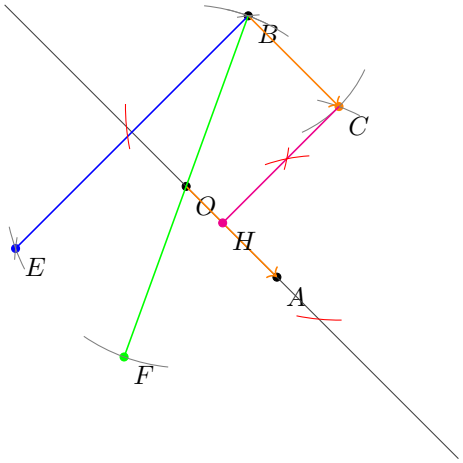
28.2 Constructions de certaines transformations \tkzShowTransformation

\tkzShowTransformation[⟨local options⟩](⟨pt1,pt2⟩) ou (⟨pt1,pt2,pt3⟩)

Ces constructions concernent les symétries orthogonales, les symétries centrales, les projections orthogonales et les translations. Plusieurs options permettent l'ajustement des constructions. L'idée de cette macro revient à **Yves Combe**

options	défaut	définition
reflection= over pt1--pt2	reflection	constructions d'une symétrie orthogonale
symmetry=center pt	reflection	constructions d'une symétrie centrale
projection=onto pt1--pt2	reflection	constructions d'une projection
translation=from pt1 to pt2	reflection	constructions d'une translation
K	1	cercle inscrit dans à un triangle
length	1	longueur d'un arc
ratio	.5	rapport entre les longueurs des arcs
gap	2	placement le point de construction
size	1	rayon d'un arc (voir bissectrice)

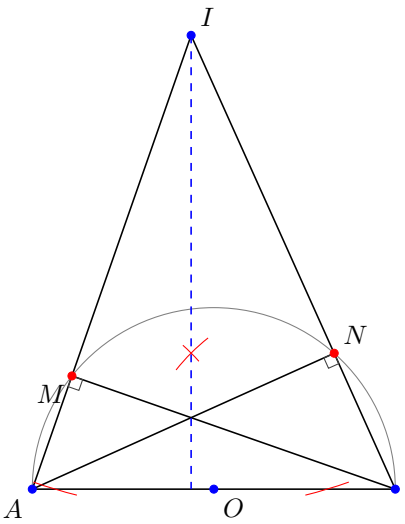
28.2.1 Exemple d'utilisation de \tkzShowTransformation




```
\begin{tikzpicture}[scale=.6]
  \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}
```

28.2.2 Autre exemple d'utilisation de \tkzShowTransformation

Vous retouverez cette figure, mais sans les traits de construction




```
\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'}
  \tkzDrawSemiCircle[diameter](A,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)
  \tkzLabelPoints(O)
  \tkzLabelPoints[above right](N,I)
  \tkzLabelPoints[below left](M,A)
\end{tikzpicture}
```

29 Différents points

29.1 \tkzDefEquiPoints

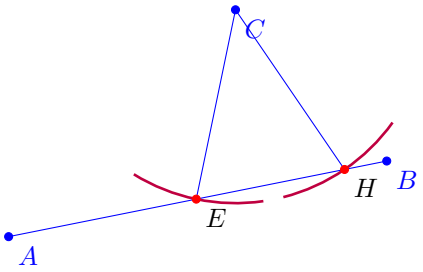
Cette macro permet d'obtenir deux points d'une droite équidistants d'un point donné.

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

arguments	défaut	définition
(pt1,pt2)	no default	liste non ordonnée de deux points

options	défaut	définition
dist	2 cm	moitié de la distance entre les deux points
from=pt	no default	point de référence
show	false	si true affiche les traces de compas
/compass/delta	0	taille des traces de compas

29.1.1 Utilisation de \tkzDefEquiPoints avec des options




```
\begin{tikzpicture}
  \tkzSetUpCompass[color=purple,line width=1pt]
  \tkzDefPoint(0,1){A}
  \tkzDefPoint(5,2){B}
  \tkzDefPoint(3,4){C}
  \tkzDefEquiPoints[from=C,dist=1,show,
      /compass/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}
```

30 Rapporteurs

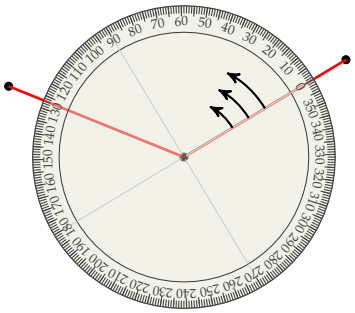
D'après une idée de Yves Combe., la macro suivante permet de dessiner un rapporteur.

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

options	défaut	définition
lw	0.4 pt	épaisseur des lignes
scale	1	ratio : permet d'ajuster la taille du rapporteur
return	false	sens indirect du cercle trigonométrique

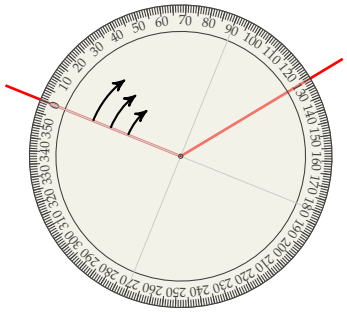
30.1 Le rapporteur circulaire

Mesure dans le sens direct



```
\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}
```

30.2 Le rapporteur circulaire, transparent et retourné



```
\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}
```

31 Des exemples

31.1 Quelques exemples intéressants

31.1.1 Triangles isocèles semblables

Ce qui suit provient de l'excellent site **Descartes et les Mathématiques**. Je n'ai pas modifié le texte et je ne suis l'auteur que de la programmation des figures.

http://debart.pagesperso-orange.fr/seconde/triangle.html

Bibliographie : Géométrie au Bac - Tangente, hors série no 8 - Exercice 11, page 11

Élisabeth Busser et Gilles Cohen : 200 nouveaux problèmes du Monde - POLE 2007

Affaire de logique n° 364 - Le Monde 17 février 2004

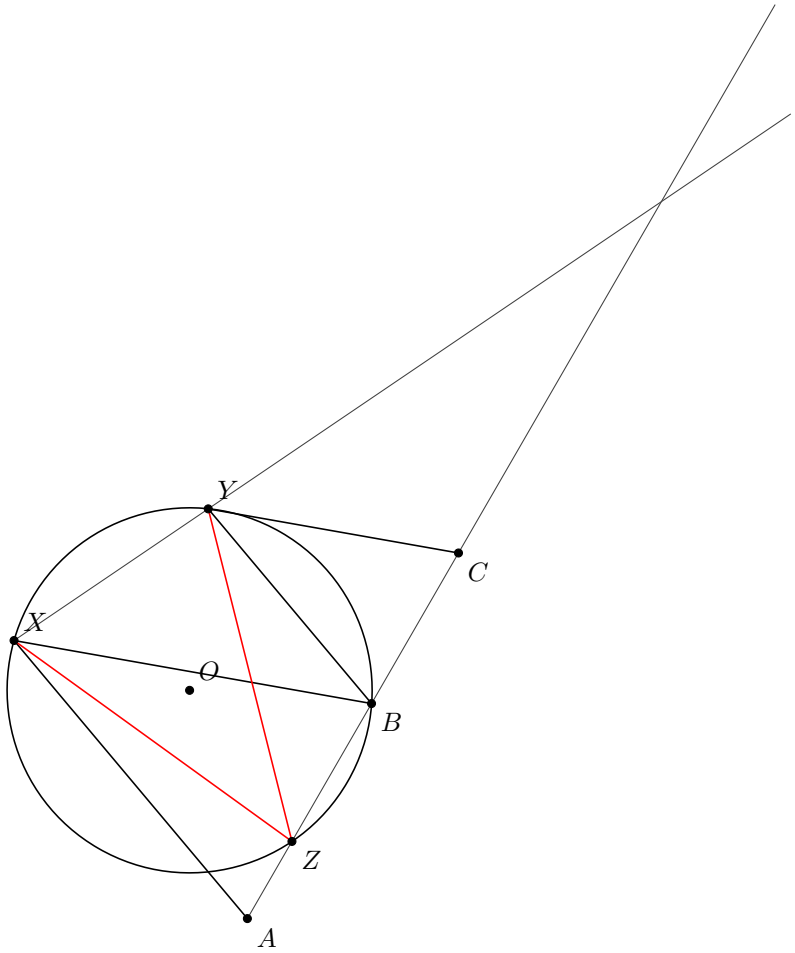
Deux énoncés ont été proposés, l'un par la revue *Tangente*, et l'autre par le journal *Le Monde*.

Rédaction de la revue Tangente : On construit deux triangles isocèles semblables AXB et BYC de sommets principaux X et Y, tels que A, B et C soient alignés et que ces triangles soient « indirect ». Soit α l'angle au sommet $\widehat{AXB} = \widehat{BYC}$. On construit ensuite un troisième triangle isocèle XZY semblable aux deux premiers, de sommet principal Z et « indirect ».
On demande de démontrer que le point Z appartient à la droite (AC).

Rédaction du Monde : On construit deux triangles isocèles semblables AXB et BYC de sommets principaux X et Y, tels que A, B et C soient alignés et que ces triangles soient « indirect ». Soit α l'angle au sommet $\widehat{AXB} = \widehat{BYC}$. Le point Z du segment [AC] est équidistant des deux sommets X et Y.
Sous quel angle voit-il ces deux sommets ?

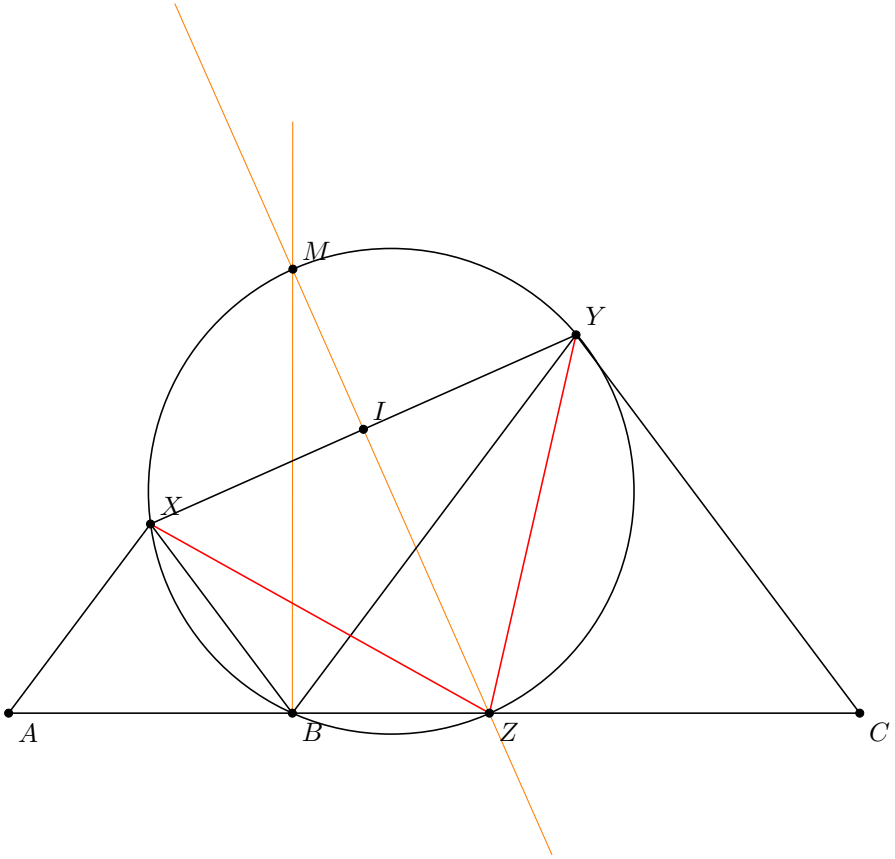
Les constructions et leurs codes associés sont sur les deux pages suivantes, mais vous pouvez chercher avant de regarder. La programmation respecte (il me semble ...), mon raisonnement dans les deux cas.

31.1.2 version revue "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}
```

31.1.3 version "Le Monde"



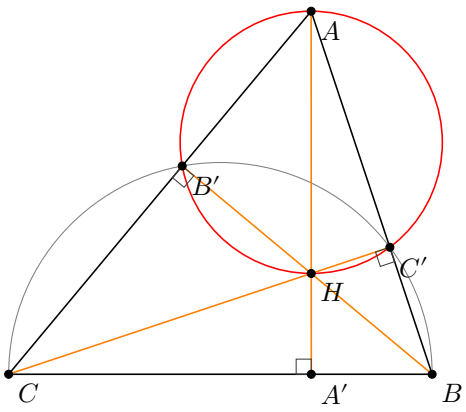

```
\begin{tikzpicture}[scale=1.25]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(3,0){B}
  \tkzDefPoint(9,0){C}
  \tkzDefPoint(1.5,2){X}
  \tkzDefPoint(6,4){Y}
  \tkzDefCircle[circum](X,Y,B) \tkzGetPoint{O}
  \tkzDefMidPoint(X,Y)                 \tkzGetPoint{I}
  \tkzDefPointWith[orthogonal](I,Y)  \tkzGetPoint{i}
  \tkzDrawLines[add = 2 and 1,color=orange](I,i)
  \tkzInterLL(I,i)(A,B)                \tkzGetPoint{Z}
  \tkzInterLC(I,i)(O,B)                \tkzGetSecondPoint{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}
```

31.1.4 Hauteurs d'un triangle

Ce qui suit provient encore de l'excellent site **Descartes et les Mathématiques**.

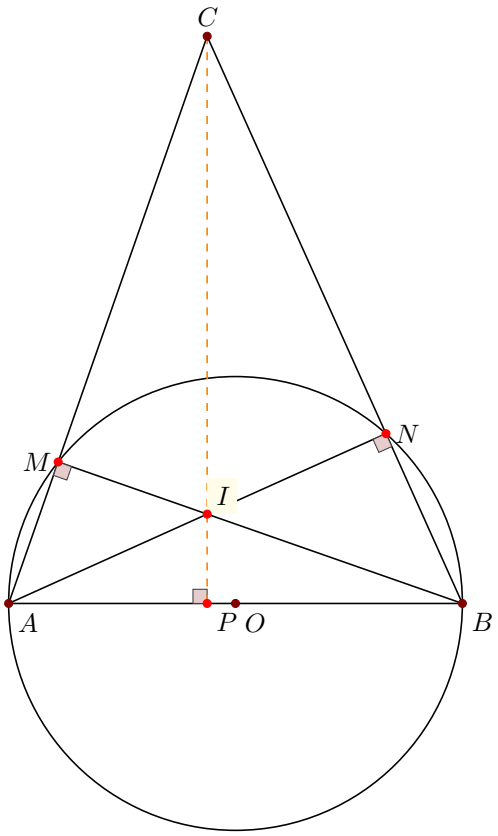
http://debart.pagesperso-orange.fr/geoplan/geometrie_triangle.html

Les trois hauteurs d'un triangle sont concourantes au même point H.




```
\begin{tikzpicture}[scale=.8]
   \tkzDefPoint(0,0){C}
   \tkzDefPoint(7,0){B}
   \tkzDefPoint(5,6){A}
   \tkzDrawPolygon(A,B,C)
   \tkzDefMidPoint(C,B)
   \tkzGetPoint{I}
   \tkzDrawArc(I,B)(C)
   \tkzInterLC(A,C)(I,B)
   \tkzGetSecondPoint{B'}
   \tkzInterLC(A,B)(I,B)
   \tkzGetFirstPoint{C'}
   \tkzInterLL(B,B')(C,C')
   \tkzGetPoint{H}
   \tkzInterLL(A,H)(C,B)
   \tkzGetPoint{A'}
     \tkzDefCircle[circum](A,B',C')
    \tkzGetPoint{O}
   \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(A,B,C,A',B',C',H)
\end{tikzpicture}
```

31.1.5 Hauteurs - autre construction

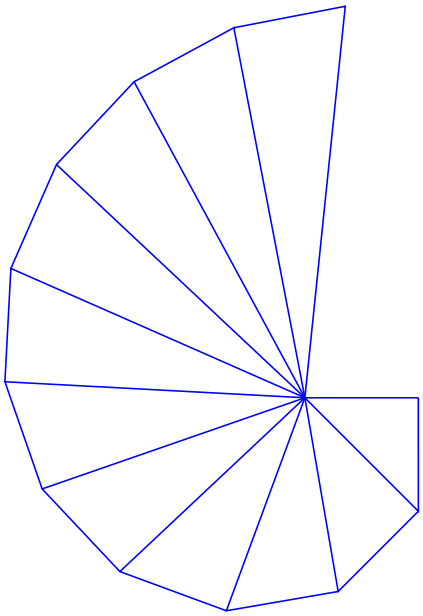



```
\begin{tikzpicture}[scale=.75]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(8,0){B}
  \tkzDefPoint(3.5,10){C}
  \tkzDefMidPoint(A,B)
  \tkzGetPoint{O}
  \tkzDefPointBy[projection=onto A--B](C)
  \tkzGetPoint{P}
  \tkzInterLC(C,A)(O,A)
  \tkzGetSecondPoint{M}
  \tkzInterLC(C,B)(O,A)
  \tkzGetFirstPoint{N}
  \tkzInterLL(B,M)(A,N)
  \tkzGetPoint{I}
  \tkzDrawCircle[diameter](A,B)
  \tkzDrawSegments(C,A C,B A,B B,M A,N)
  \tkzMarkRightAngles[fill=Maroon!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[fill=fondpaille,above right](I){$I$}
  \tkzDrawPoints[color=red](M,N,P,I)
  \tkzDrawPoints[color=Maroon](O,A,B,C)
\end{tikzpicture}
```

31.2 Different authors

31.2.1 Square root of the integers

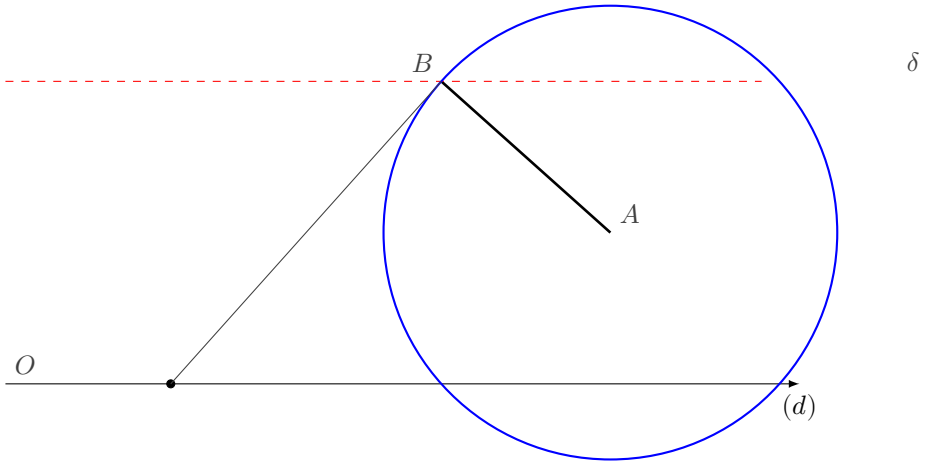
How to get 1, $\sqrt{2}$, $\sqrt{3}$ with a rule and a compass.



```
\begin{tikzpicture}[scale=1.5]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(1,0){a0}
   \tkzDrawSegment[blue](O,a0)
  \foreach \i [count=\j] in {0,...,10}{%
    \tkzDefPointWith[orthogonal normed](a\i,O)
    \tkzGetPoint{a\j}
    \tkzDrawPolySeg[color=blue](a\i,a\j,O)}
 \end{tikzpicture}
```

31.2.2 Circle and tangent

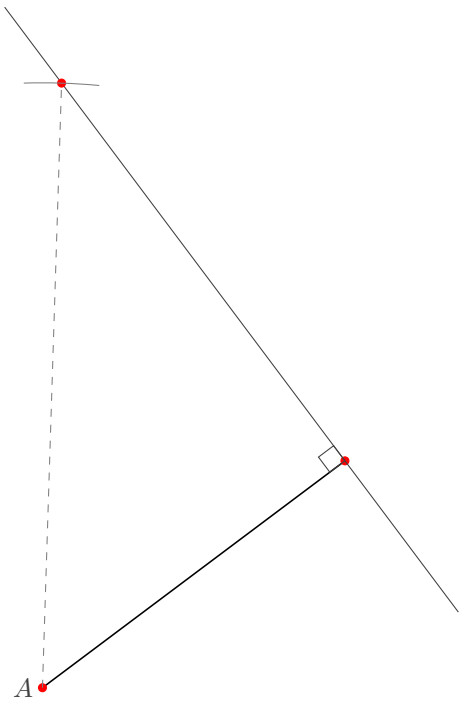
We have a point A $(8,2)$, a circle with center A and radius=3cm and a line δ $y = 4$. The line intercepts the circle at B. We want to draw the tangent at the circle in B.



```
\begin{tikzpicture}
  \edef\alphaR{\fpeval{asin(2/3)}}
  \edef\xB{8-3*cos(\alphaR)}
  \tkzDrawX[noticks,label=$(d)$]
  \tkzDefPoint["$A$" above right](8,2){A}
  \tkzDefPoint[color=red,"$O$" above right](0,0){O}
  \tkzDefPoint["$B$" above left](\xB,4){B}
  \tkzDefLine[orthogonal=through B](A,B) \tkzGetPoint{b}
  \tkzDefPoint(1,0){i}
  \tkzInterLL(B,b)(O,i) \tkzGetPoint{B'}
  \tkzDrawSegment[line width=1pt](A,B)
  \tkzHLine[color=red,style=dashed]{4}
  \tkzText[above](12,4){$\delta$}
  \tkzDrawCircle[R,color=blue,line width=.8pt](A,3 cm)
  \tkzDrawPoint(B')
  \tkzDrawLine(B,B')
 \end{tikzpicture}
```

31.2.3 About right triangle

We have a segment $[AB]$ and we want to determine a point C such as $AC = 8cm$ and ABC is a right triangle in B.

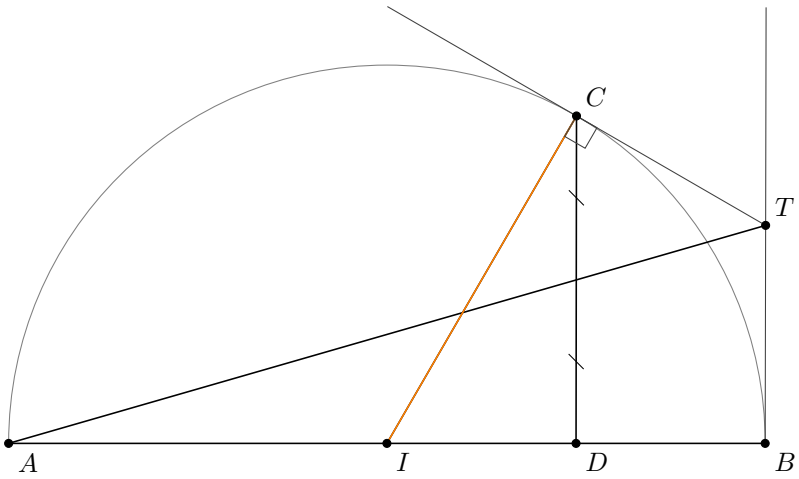



```
\begin{tikzpicture}
  \tkzDefPoint["$A$" left](2,1){A}
  \tkzDefPoint(6,4){B}
  \tkzDrawSegment(A,B)
  \tkzDrawPoint[color=red](A)
  \tkzDrawPoint[color=red](B)
  \tkzDefPointWith[orthogonal,K=-1](B,A)
  \tkzDrawLine[add = .5 and .5](B,tkzPointResult)
  \tkzInterLC[R](B,tkzPointResult)(A,8 cm)
  \tkzGetPoints{C}{J}
  \tkzDrawPoint[color=red](C)
  \tkzCompass(A,C)
  \tkzMarkRightAngle(A,B,C)
  \tkzDrawLine[color=gray,style=dashed](A,C)
\end{tikzpicture}
```

31.2.4 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. An other tangent line at a point, C, on the semicircle is drawn. We project the point C on the segment$[AB]$ on a point D . The two tangent lines intersect at the point T.

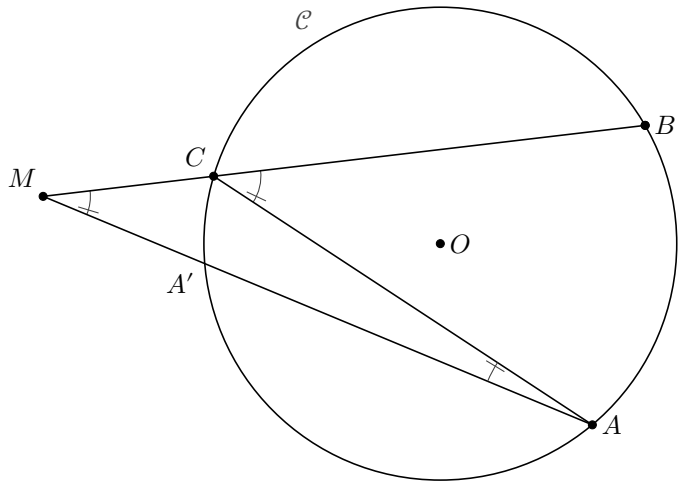
Prove that the line (AT) bisects (CD)




```
\begin{tikzpicture}[scale=1.25]
  \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 cm) \tkzGetFirstPoint{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[color=orange](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}
```

31.2.5 Exemple : Dimitris Kapeta

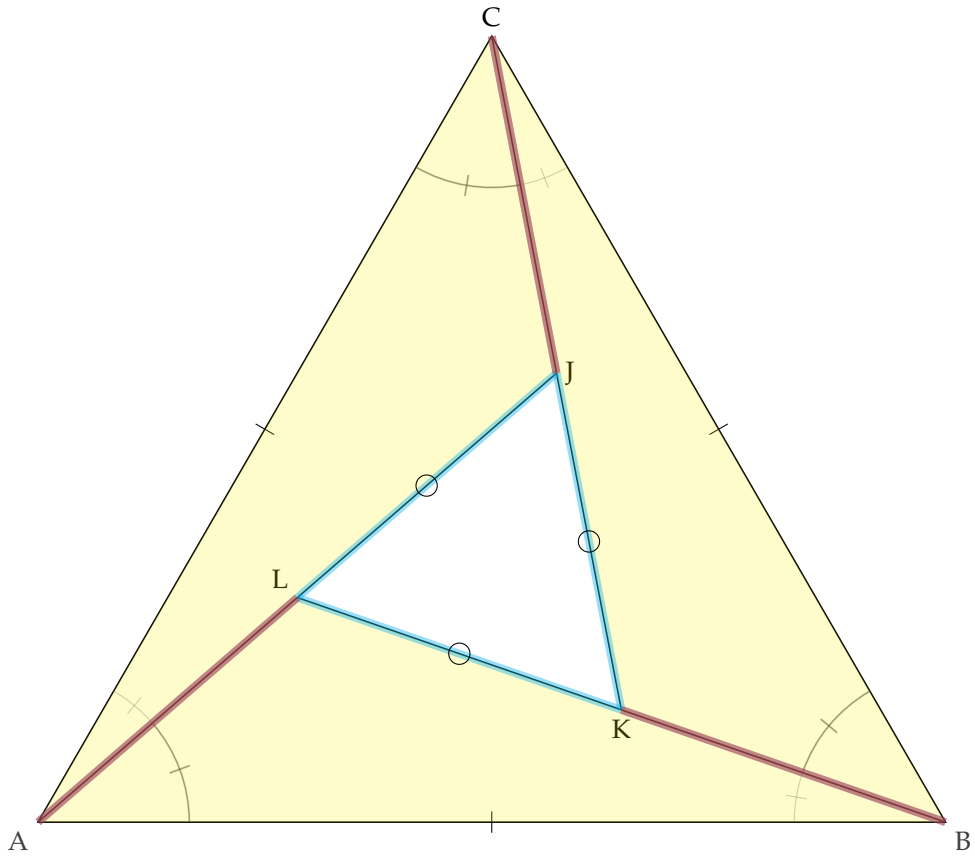
You need in this example to use **mkpos=.2** with **\tkzMarkAngle** because the measure of $\widehat{CAM}$ is too small. Another possiblity is to use **\tkzFillAngle**.



```
\begin{tikzpicture}[scale=1.25]
  \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(M,B)(O,N) \tkzGetFirstPoint{C}
  \tkzInterLC(M,A)(O,N) \tkzGetSecondPoint{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)
  \tkzDrawCircle(O,N)
  \tkzLabelCircle[above left](O,N)(120){$\mathcal{C}$}
  \tkzMarkAngle[mkpos=.2, size=1.2](C,A,M)
  \tkzDrawPoints(O, A, B, M, B, C)
  \tkzLabelPoints[right](O,A,B)
  \tkzLabelPoints[above left](M,C)
  \tkzLabelPoint[below left](A'){$A'$}
\end{tikzpicture}
```

31.2.6 Example : John Kitzmiller

Prove $\triangle LKJ$ is equilateral

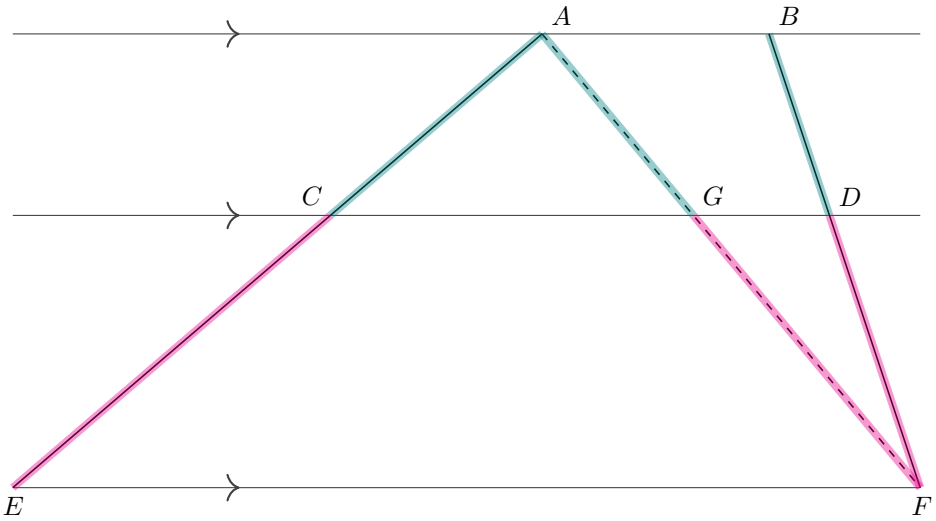


```
\begin{tikzpicture}[scale=2]
  \tkzDefPoint[label=below left:A](0,0){A}
  \tkzDefPoint[label=below right:B](6,0){B}
  \tkzDefTriangle[equilateral](A,B) \tkzGetPoint{C}
  \tkzMarkSegments[mark=|](A,B A,C B,C)
  \tkzDefBarycentricPoint(A=1,B=2) \tkzGetPoint{C'}
  \tkzDefBarycentricPoint(A=2,C=1) \tkzGetPoint{B'}
  \tkzDefBarycentricPoint(C=2,B=1) \tkzGetPoint{A'}
  \tkzInterLL(A,A')(C,C') \tkzGetPoint{J}
  \tkzInterLL(C,C')(B,B') \tkzGetPoint{K}
  \tkzInterLL(B,B')(A,A') \tkzGetPoint{L}
  \tkzLabelPoint[above](C){C}
  \tkzDrawPolygon(A,B,C) \tkzDrawSegments(A,J B,L C,K)
  \tkzMarkAngles[fill= orange,size=1cm,opacity=.3](J,A,C K,C,B L,B,A)
  \tkzLabelPoint[right](J){J}
  \tkzLabelPoint[below](K){K}
  \tkzLabelPoint[above left](L){L}
  \tkzMarkAngles[fill=orange, opacity=.3,thick,size=1,](A,C,J C,B,K B,A,L)
  \tkzMarkAngles[fill=green, size=1, opacity=.5](A,C,J C,B,K B,A,L)
  \tkzFillPolygon[color=yellow, opacity=.2](J,A,C)
  \tkzFillPolygon[color=yellow, opacity=.2](K,B,C)
  \tkzFillPolygon[color=yellow, opacity=.2](L,A,B)
  \tkzDrawSegments[line width=3pt,color=cyan,opacity=0.4](A,J C,K B,L)
  \tkzDrawSegments[line width=3pt,color=red,opacity=0.4](A,L B,K C,J)
  \tkzMarkSegments[mark=o](J,K K,L L,J)
\end{tikzpicture}
```

31.2.7 Exemple : John Kitzmiller

Prove $\dfrac{AC}{CE} = \dfrac{BD}{DF}$

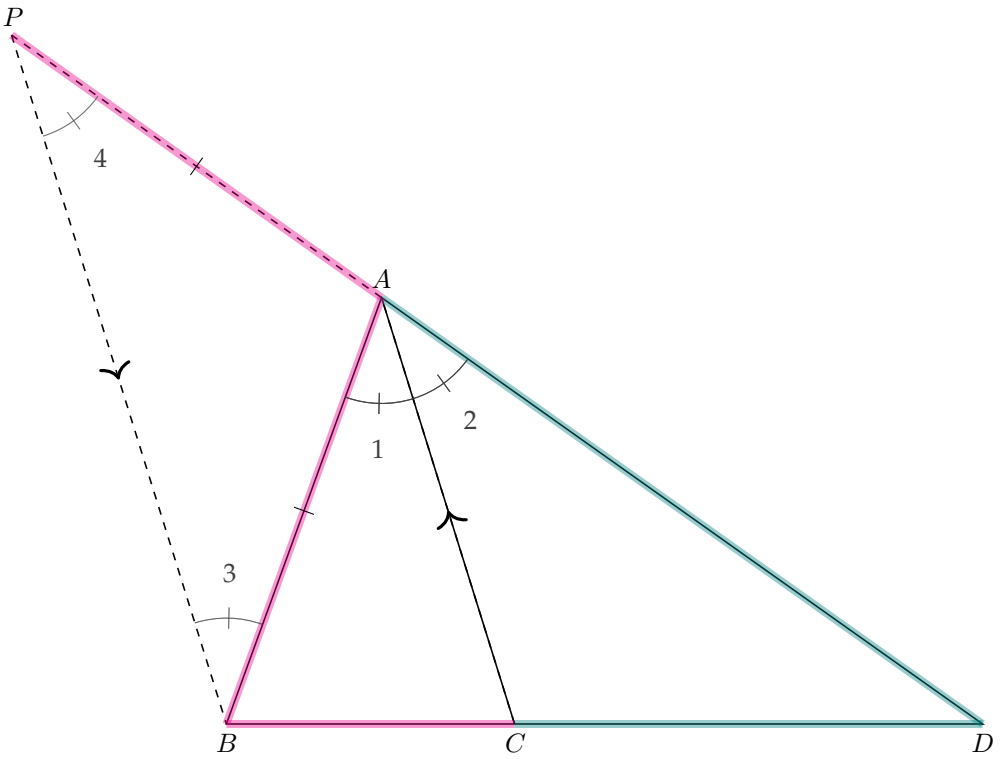
Another interesting example from John, you can see how to use some extra options like **decoration** and **postaction** from Ti*k*Z with **tkz-euclide**.



```
\begin{tikzpicture}[scale=2,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}
```

31.2.8 Exemple : John Kitzmiller

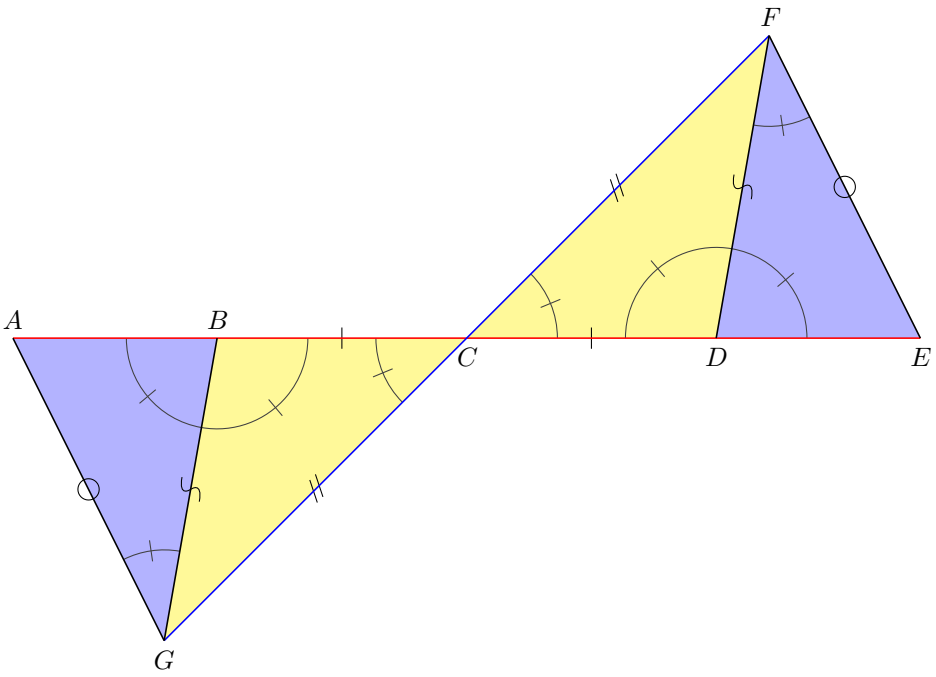
Prove $\dfrac{BC}{CD} = \dfrac{AB}{AD}$ (Angle Bisector)



```
\begin{tikzpicture}[scale=2]
  \tkzDefPoints{0/0/B, 5/0/D}       \tkzDefPoint(70:3){A}
  \tkzDrawPolygon(B,D,A)
  \tkzDefLine[bisector](B,A,D)      \tkzGetPoint{a}
  \tkzInterLL(A,a)(B,D)             \tkzGetPoint{C}
  \tkzDefLine[parallel=through B](A,C) \tkzGetPoint{b}
  \tkzInterLL(A,D)(B,b)             \tkzGetPoint{P}
  \begin{scope}[decoration={markings,
   mark=at position .5 with {\arrow[scale=2]{>}}}]
   \tkzDrawSegments[postaction={decorate},dashed](C,A P,B)
  \end{scope}
  \tkzDrawSegment(A,C) \tkzDrawSegment[style=dashed](A,P)
  \tkzLabelPoints[below](B,C,D) \tkzLabelPoints[above](A,P)
  \tkzDrawSegments[color=magenta, line width=3pt, opacity=0.4](B,C P,A)
  \tkzDrawSegments[color=teal,    line width=3pt, opacity=0.4](C,D A,D)
  \tkzDrawSegments[color=magenta, line width=3pt, opacity=0.4](A,B)
  \tkzMarkAngles[size=0.7](B,A,C C,A,D)
  \tkzMarkAngles[size=0.7, fill=green,   opacity=0.5](B,A,C A,B,P)
  \tkzMarkAngles[size=0.7, fill=yellow, opacity=0.3](B,P,A C,A,D)
  \tkzMarkAngles[size=0.7, fill=green,   opacity=0.6](B,A,C A,B,P B,P,A C,A,D)
  \tkzLabelAngle[pos=1](B,A,C){1}   \tkzLabelAngle[pos=1](C,A,D){2}
  \tkzLabelAngle[pos=1](A,B,P){3}    \tkzLabelAngle[pos=1](B,P,A){4}
  \tkzMarkSegments[mark=|](A,B A,P)
\end{tikzpicture}
```

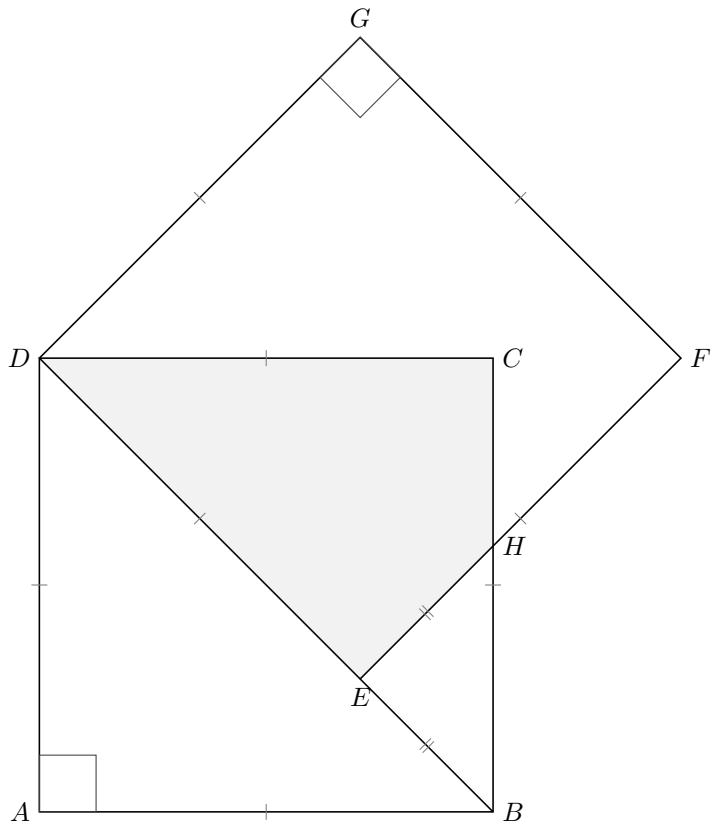
31.2.9 Exemple : author John Kitzmiller

Prove $\overline{AG} \cong \overline{EF}$ (Detour)



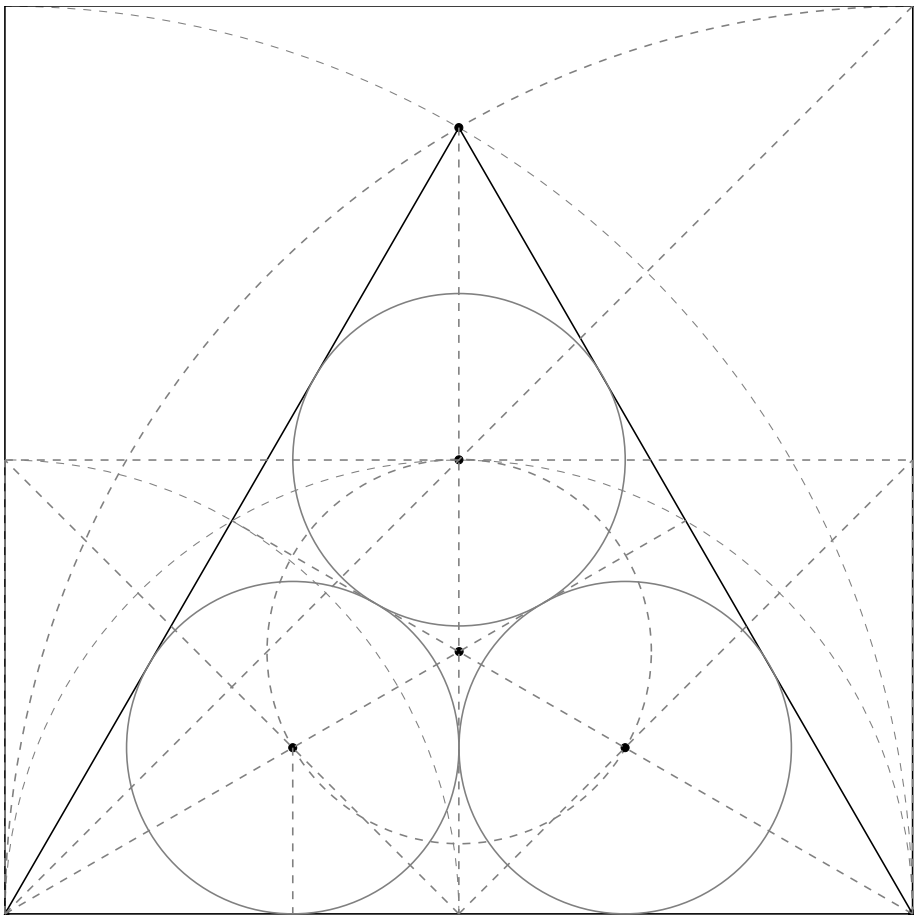
```
\begin{tikzpicture}[scale=2]
  \tkzDefPoint(0,3){A}    \tkzDefPoint(6,3){E}  \tkzDefPoint(1.35,3){B}
  \tkzDefPoint(4.65,3){D} \tkzDefPoint(1,1){G}  \tkzDefPoint(5,5){F}
  \tkzDefMidPoint(A,E)    \tkzGetPoint{C}
  \tkzFillPolygon[yellow, opacity=0.4](B,G,C)
  \tkzFillPolygon[yellow, opacity=0.4](D,F,C)
  \tkzFillPolygon[blue, opacity=0.3](A,B,G)
  \tkzFillPolygon[blue, opacity=0.3](E,D,F)
  \tkzMarkAngles[size=0.6,fill=green](B,G,A D,F,E)
  \tkzMarkAngles[size=0.6,fill=orange](B,C,G D,C,F)
  \tkzMarkAngles[size=0.6,fill=yellow](G,B,C F,D,C)
  \tkzMarkAngles[size=0.6,fill=red](A,B,G E,D,F)
  \tkzMarkSegments[mark=|](B,C D,C)  \tkzMarkSegments[mark=s||](G,C F,C)
  \tkzMarkSegments[mark=o](A,G E,F)  \tkzMarkSegments[mark=s](B,G D,F)
  \tkzDrawSegment[color=red](A,E)
  \tkzDrawSegment[color=blue](F,G)
  \tkzDrawSegments(A,G G,B E,F F,D)
  \tkzLabelPoints[below](C,D,E,G)  \tkzLabelPoints[above](A,B,F)
\end{tikzpicture}
```

31.2.10 Idea from Indonesia



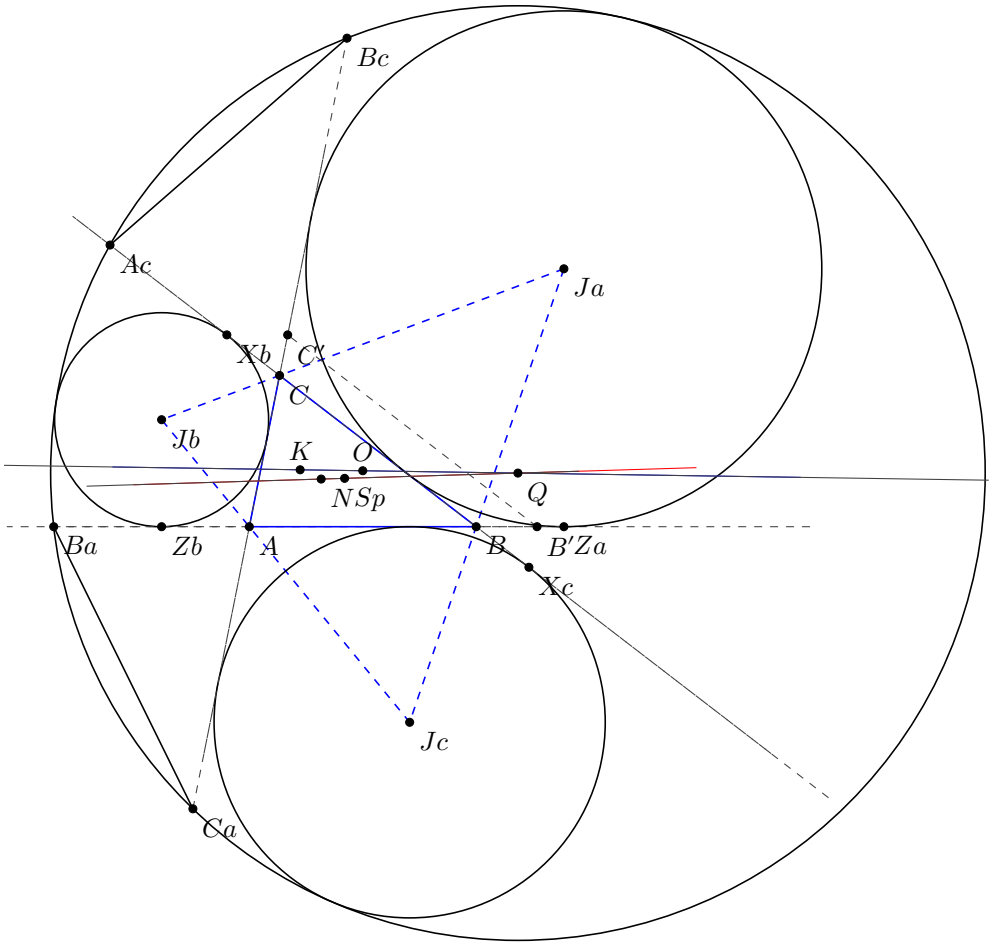
```
\begin{tikzpicture}[scale=3]
   \tkzDefPoints{0/0/A,2/0/B}
   \tkzDefSquare(A,B) \tkzGetPoints{C}{D}
   \tkzDefPointBy[rotation=center D angle 45](C)\tkzGetPoint{G}
   \tkzDefSquare(G,D)\tkzGetPoints{E}{F}
   \tkzInterLL(B,C)(E,F)\tkzGetPoint{H}
   \tkzFillPolygon[gray!10](D,E,H,C,D)
   \tkzDrawPolygon(A,...,D)\tkzDrawPolygon(D,...,G)
   \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}
```

31.2.11 Three circles



```
\begin{tikzpicture}[scale=1.5]
  \tkzDefPoints{0/0/A,8/0/B,0/4/a,8/4/b,8/8/c}
  \tkzDefTriangle[equilateral](A,B) \tkzGetPoint{C}
  \tkzDrawPolygon(A,B,C)
  \tkzDefSquare(A,B) \tkzGetPoints{D}{E}
  \tkzClipBB
  \tkzDefMidPoint(A,B) \tkzGetPoint{M}
  \tkzDefMidPoint(B,C) \tkzGetPoint{N}
  \tkzDefMidPoint(A,C) \tkzGetPoint{P}
  \tkzDrawSemiCircle[gray,dashed](M,B)
  \tkzDrawSemiCircle[gray,dashed](A,M)
  \tkzDrawSemiCircle[gray,dashed](A,B)
  \tkzDrawCircle[gray,dashed](B,A)
  \tkzInterLL(A,N)(M,a) \tkzGetPoint{Ia}
  \tkzDefPointBy[projection = onto A--B](Ia)
  \tkzGetPoint{ha}
  \tkzDrawCircle[gray](Ia,ha)
  \tkzInterLL(B,P)(M,b) \tkzGetPoint{Ib}
  \tkzDefPointBy[projection = onto A--B](Ib)
  \tkzGetPoint{hb}
  \tkzDrawCircle[gray](Ib,hb)
  \tkzInterLL(A,c)(M,C) \tkzGetPoint{Ic}
  \tkzDefPointBy[projection = onto A--C](Ic)
  \tkzGetPoint{hc}
  \tkzDrawCircle[gray](Ic,hc)
  \tkzInterLL(A,Ia)(B,Ib) \tkzGetPoint{G}
  \tkzDrawCircle[gray,dashed](G,Ia)
  \tkzDrawPolySeg(A,E,D,B)
  \tkzDrawPoints(A,B,C)
  \tkzDrawPoints(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}
```

31.2.12 "The" Circle of APOLLONIUS



```
\begin{tikzpicture}[scale=.5]
\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[spieker](A,B,C)      \tkzGetPoint{Sp}
\tkzDefExCircle(A,B,C)     \tkzGetPoint{Jb}
\tkzDefExCircle(C,A,B)     \tkzGetPoint{Ja}
\tkzDefExCircle(B,C,A)     \tkzGetPoint{Jc}
\tkzDefPointBy[projection=onto B--C ](Jc)   \tkzGetPoint{Xc}
\tkzDefPointBy[projection=onto B--C ](Jb)   \tkzGetPoint{Xb}
\tkzDefPointBy[projection=onto A--B ](Ja)   \tkzGetPoint{Za}
\tkzDefPointBy[projection=onto A--B ](Jb)   \tkzGetPoint{Zb}
\tkzDefLine[parallel=through Xc](A,C)       \tkzGetPoint{X'c}
\tkzDefLine[parallel=through Xb](A,B)       \tkzGetPoint{X'b}
\tkzDefLine[parallel=through Za](C,A)       \tkzGetPoint{Z'a}
\tkzDefLine[parallel=through Zb](C,B)       \tkzGetPoint{Z'b}
\tkzInterLL(Xc,X'c)(A,B)                    \tkzGetPoint{B'}
\tkzInterLL(Xb,X'b)(A,C)                    \tkzGetPoint{C'}
\tkzInterLL(Za,Z'a)(C,B)                    \tkzGetPoint{A''}
\tkzInterLL(Zb,Z'b)(C,A)                    \tkzGetPoint{B''}
\tkzDefPointBy[reflection= over Jc--Jb](B') \tkzGetPoint{Ca}
\tkzDefPointBy[reflection= over Jc--Jb](C') \tkzGetPoint{Ba}
\tkzDefPointBy[reflection= over Ja--Jb](A'')\tkzGetPoint{Bc}
\tkzDefPointBy[reflection= over Ja--Jb](B'')\tkzGetPoint{Ac}
\tkzDefCircle[circum](Ac,Ca,Ba)             \tkzGetPoint{Q}
\tkzDrawCircle[circum](Ac,Ca,Ba)
\tkzDefPointWith[linear,K=1.1](Q,Ac)        \tkzGetPoint{nAc}
\tkzClipCircle[through](Q,nAc)
\tkzDrawLines[add=1.5 and 1.5,dashed](A,B B,C A,C)
\tkzDrawPolygon[color=blue](A,B,C)
\tkzDrawPolygon[dashed,color=blue](Ja,Jb,Jc)
\tkzDrawCircles[ex](A,B,C B,C,A C,A,B)
\tkzDrawLines[add=0 and 0,dashed](Ca,Bc B,Za A,Ba B',C')
\tkzDrawLine[add=1 and 1,dashed](Xb,Xc)
\tkzDrawLine[add=7 and 3,blue](O,K)
\tkzDrawLine[add=8 and 15,red](N,Sp)
\tkzDrawLines[add=10 and 10](K,O N,Sp)
\tkzDrawSegments(Ba,Ca Bc,Ac)
\tkzDrawPoints(A,B,C,N,Ja,Jb,Jc,Xb,Xc,B',C',Za,Zb,Ba,Ca,Bc,Ac,Q,Sp,K,O)
\tkzLabelPoints(A,B,C,N,Ja,Jb,Jc,Xb,Xc,B',C',Za,Zb,Ba,Ca,Bc,Ac,Q,Sp)
\tkzLabelPoints[above](K,O)
\end{tikzpicture}
```

32 FAQ

32.1 Erreurs les plus fréquentes

Je me base pour le moment sur les miennes, car ayant changé plusieurs fois de syntaxes, j'ai commis un certain nombre d'erreurs. Cette section est amenée à se développer.

— **\tkzDrawPoint(A,B)** alors qu'il faut **\tkzDrawPoints**

— **\tkzGetPoint(A)** Quand on définit un objet, il faut utiliser des accolades et non des parenthèses, il faut donc écrire : **\tkzGetPoint{A}**

— **\tkzGetPoint{A}** à la place de **\tkzGetFirstPoint{A}**. Quant une macro donne deux points comme résultats, soit on récupère ces points à l'aide de **\tkzGetPoints{A}{B}**, soit on ne récupère que l'un des deux points, à l'aide **\tkzGetFirstPoint{A}** ou bien de **\tkzGetSecondPoint{A}**. Ces deux points peuvent être utilisés avec comme référence **tkzFirstPointResult** ou **tkzSecondPointResult**. Il est possible qu'un troisième point soit donné sous la référence **tkzPointResult**

— **\tkzDrawSegment(A,B A,C)** alors qu'il faut **\tkzDrawSegments**. Il est possible de n'utiliser que les versions avec un « s » mais c'est moins efficace !

— Mélange option et arguments ; toutes les macros qui utilisent un cercle ont besoin de connaître le rayon de celui-ci. Si le rayon est donné par une mesure alors l'option comprend un **R**.

— **\tkzDrawSegments[color = gray,style=dashed]{B,B' C,C'}** est une erreur. Seules, les macros qui définissent un objet utilisent des accolades.

— Les angles sont donnés en degrés plus rarement en radians.

— Si une erreur survient dans un calcul lors d'un passage de paramètres, alors il est préférable de faire ces calculs avant d'appeler la macro.

— Ne pas mélanger la syntaxe de **pgfmath** et celle de **xfp**. J'ai choisi souvent **xfp** mais si vous préférez pgfmath alors effectuez vos calculs avant le passage de paramètres.

— Usage de **\tkzClip** : Afin d'avoir des résultats précis, j'ai évité de passer par des vecteurs normalisés. L'avantage de la normalisation est de contrôler la dimension des objets manipulés, le désavantage est qu'avec TeX, cela implique des imprécisions. Ces imprécisions sont souvent minimes, de l'ordre du millième, mais entraînent des catastrophes si le dessin est agrandi. Ne pas normaliser implique que certains points se trouvent bien loin de la zone de travail et **\tkzClip** permet de réduire la taille du dessin.

— Une erreur se produit si vous utilisez la macro **\tkzDrawAngle** avec un angle trop petit. L'erreur est produite par la librairie **decoration** quand on veut placer une marque sur un arc. Même si la marque est absente, l'erreur, elle, reste présente. Il est possible de contourner cette difficulté avec l'option **mkpos=.2** par exemple, qui placera la marque avant l'arc. Une autre possibilité est d'utiliser la macro **\tkzFillAngle**

Index