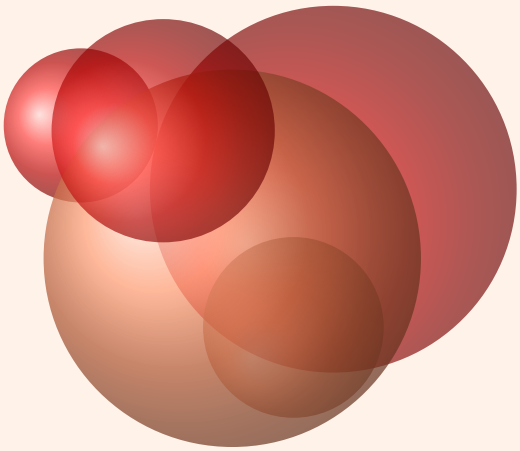
tkz-euclide 1.13 c

AlterMundus



Alain Matthes

20 janvier 2011

http://altermundus.fr http://altermundus.com

tkz-euclide

AlterMundus

Alain Matthes

*Le package **tkz-euclide.sty** est un ensemble de macros spécialisées permettant de construire des objets géométriques en 2D dans un plan muni d'un repère. Il est construit au-dessus de PGF et son interface TikZ. Ce document fournit les définitions des différentes macros ainsi que des exemples dont la complexité est graduée. **tkz-euclide.sty** remplace **tkz-2d.sty** dont le code n'est plus maintenu. Ce package nécessite la version 2.1 de **TikZ**.*

☞ Je souhaite remercier **Till Tantau** pour avoir créé le merveilleux outil Ti*k*Z, ainsi que **Michel Bovani** pour **fourier**, dont l'association avec **utopia** est excellente.

☞ 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 de nombreux exemples sur mes sites : altermundus.com ou 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

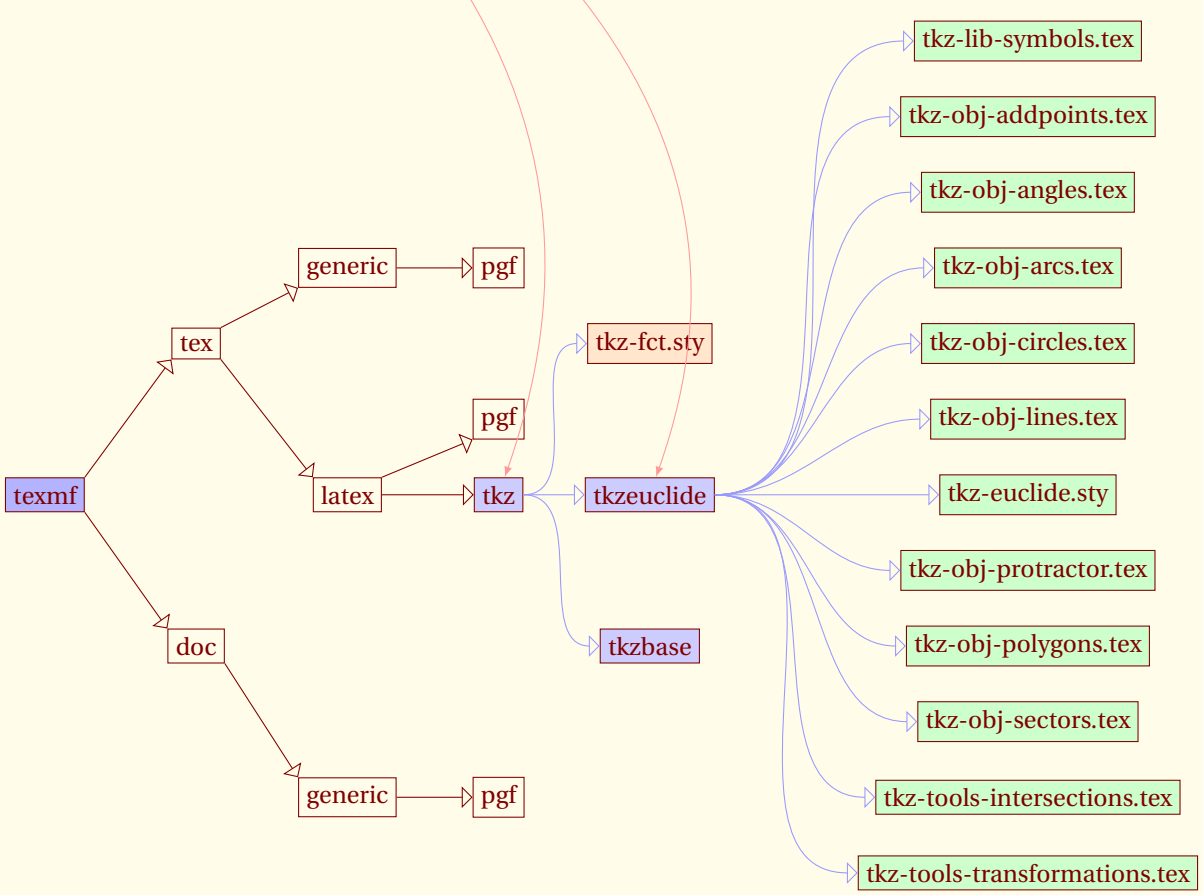
SECTION 1

Installation

Lorsque vous lirez ce document, il est possible que **tkz-euclide** soit présent sur le serveur du **CTAN**[1] alors **tlmgr** vous permettra de l'installer. Si **tkz-euclide** ne fait pas encore partie de votre distribution, cette section vous montre comment l'installer, elle est aussi nécessaire si vous avez envie d'installer une version beta ou personnalisée de **tkz-euclide**. Si le package est présent sur le serveur du **CTAN** et que vous n'utilisez pas **tlmgr**, je vous conseille de la télécharger à partir de ce serveur, sinon vous le trouverez sur mon site. Pour distinguer les anciennes versions de la nouvelle, j'ai repris la numérotation à 1.00 et j'ai ajouté « c »[2] . Vous allez donc installer la version **1.13 c**.

Le plus simple est de créer un dossier tkz [3] avec comme chemin : texmf/tex/latex/tkz .

1. Après l'avoir décompressé, placez le dossier tkzeuclide dans le dossier tkz. Le dossier **tkzbase** doit se trouver aussi dans le dossier **tkz**.



1. **tkz-euclide** ne fait pas encore partie de **TeXLive**
2. pour CTAN
3. ou bien un autre nom

Il est nécessaire que **tkz-base** soit aussi installé. Le plus simple est d'installer **tkz** complètement.

2. Ouvrir un terminal, puis faire `sudo texhash` si nécessaire.
3. Vérifier que **fp**, **numprint** et **tikz 2.10** sont installés car ils sont obligatoires, pour le bon fonctionnement de **tkz-euclide**.

Voici les chemins du dossier tkz sur mes deux ordinateurs :

– sous OS X **/Users/ego/Library/texmf** ;

– sous Ubuntu **/home/ego/texmf** .

Je suppose que si vous mettez vos packages ailleurs, vous savez pourquoi !

remarque : l'installation proposée n'est valable que pour un utilisateur.

1.1 Avec MikTeX sous Windows XP

Je ne connais pas grand-chose à ce système, mais un utilisateur de mes packages **Wolfgang Buechel** a eu la gentillesse de me faire parvenir ce qui suit :

Pour ajouter **tkzeuclide** à MiKTeX[4] :

– ajouter un dossier **tkz** dans le dossier `[MiKTeX-dir]/tex/latex`

– copier **tkzeuclide** et tous les fichiers présents dans le dossier **tkz**,

– mettre à jour MiKTeX, pour cela dans shell DOS lancer la commande `mktexlsr -u`
ou bien encore, choisir `Start/Programs/Miktex/Settings/General`
puis appuyer sur le bouton `Refresh FNDB`.

1.2 Liste des fichiers des dossiers tkzbase et tkzeuclide

Dans le dossier **base** :

– **tkz-base.cfg**
– **tkz-base.sty**
– **tkz-obj-marks.tex**
– **tkz-obj-points.tex**
– **tkz-obj-segments.tex**
– **tkz-tools-arith.tex**
– **tkz-tools-base.tex**
– **tkz-tools-math.tex**
– **tkz-tools-misc.tex**
– **tkz-tools-utilities.tex**

Dans le dossier **euclide** :

– **tkz-euclide.sty**
– **tkz-lib-symbols.tex**
– **tkz-obj-addpoints.tex**
– **tkz-obj-angles.tex**
– **tkz-obj-arcs.tex**
– **tkz-obj-circles.tex**
– **tkz-obj-lines.tex**

4. Essai réalisé avec la version **2.7**

- **tkz-obj-protractor.tex**
- **tkz-obj-polygons.tex**
- **tkz-obj-sectors.tex**
- **tkz-obj-vectors.tex**
- **tkz-tools-intersections.tex**
- **tkz-tools-transformations.tex**

1.3 Chargement des fichiers avec usetkzobj

Il n'était pas nécessaire de tout charger en une seule fois, seuls les fichiers indispensables sont installés. **\usepackage{tkz-base}** charge tous les fichiers présents dans le dossier **tkzbase** ; en particulier, les fichiers "objets" **tkz-obj-points.tex** et **tkz-obj-segments.tex** et **tkz-obj-marks.tex**. **\usepackage{tkz-euclide}** va ajouter des outils indispensables, mais vous devrez indiquer quels objets vous seront utiles. Pour tout charger, vous pouvez écrire : **\usetkzobj{all}** mais sinon vous pouvez demander : **\usetkzobj{cercles, arcs, protractor}**.

SECTION 2

Présentation

2.1 À propos de TikZ et que peut apporter tkz-euclide.sty ?

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 en 2d et en particulier pour créer des dessins liés à la géométrie euclidienne. Quels arguments peuvent intervenir ?

1. Certains utilisateurs n'ont pas envie d'apprendre quoi que ce soit sur **TikZ**, 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 **TikZ**, mais ressemble davantage à celle de LaTeX.
2. Les noms des macros ont une signification plus mathématique.
3. La grande différence avec **TikZ** est qu'il est possible d'utiliser des grandes valeurs ainsi que des très petites, car la majorité des calculs sont faits à l'aide de **fp.sty**. C'est plus lent, mais nettement 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.
5. Des exemples de constructions géométriques sont fournies et peuvent être utiles au débutant.
6. Et pour terminer, cela peut être une approche en douceur de l'utilisation de **TikZ** par l'intermédiaire des options. Dans cette nouvelle version, j'ai essayé que les options de **TikZ** soient pratiquement toujours disponibles.

Je vous encourage toutefois à étudier **TikZ**. En effet, l'utilisation de **tkz-euclide.sty** fait perdre la notion de **path**. 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.

2.2 À propos de tkz-euclide

Le but est donc de créer des dessins en 2D sur une page à priori A4, mais si je me suis préoccupé d'utiliser une surface inférieure, j'avoue ne pas avoir testé la possibilité de travailler sur une page de taille supérieure.

Avec **tkz-euclide**, l'unité est le centimètre. Si votre travail ne concerne que de la géométrie classique, je vous conseille de conserver cette unité.

*Pourquoi **tkz-2d** disparait-il ?*

Je n'étais pas content de la syntaxe qui était confuse, je n'avais pas utilisé pgf 2.00 et surtout j'ai généralisé l'utilisation de **fp.sty**.

SECTION 3

Syntaxe

Quelques mots sur la syntaxe.

Les accolades sont réservés pour la création d'objets et les parenthèses ne sont utilisées que pour des objets, déjà existants :

\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)** crée un point, qui est ici, le milieu d'un segment. Le point est nommé **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 :

- **\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}
```

3.1 Notions générales

Le principe est de définir des points en utilisant des coordonnées cartésiennes ou des coordonnées polaires et même des coordonnées barycentriques.

Ensuite, il est possible 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 **fp.sty** 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 quatre étapes :

1. Définir les points dont les coordonnées sont connues ou bien calculables.
2. Création de nouveaux points à l'aide de méthodes (intersection, transformation,etc.).

3. Tracés des objets dans un ordre choisi.
4. Placement des labels.

Les coordonnées peuvent être obtenues à l'aide de calculs en utilisant pgfmath, fp ou encore TEX. Toutes les macros n'acceptent pas que les calculs soient faits pendant leurs assignations. Après avoir toléré ce comportement, je l'ai abandonné afin de laisser plus de souplesse à l'utilisateur. **fp.sty** est plus précis **pgfmath**, plus rapide aussi tout dépend des constructions demandées.

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, donne la mesure d'un angle 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**. Des macros sont fournies pour nommer différemment ces résultats et les conserver. Il pourrait paraître plus simple de donner un paramètre supplémentaire à la macro pour nommer directement le résultat, mais par exemple, on peut n'avoir besoin que d'un point sur deux après une intersection, une macro peut définir trois résultats un angle , une longueur et un point. Ensuite il est facile à l'utilisateur de créer des macros qui feront tout cela d'un seul coup si cela est nécessaire.

\tkzDefPoint utilise des accolades ainsi que les macros créant des labels. Il en est de même des transformations quand elles agissent sur une liste de points.

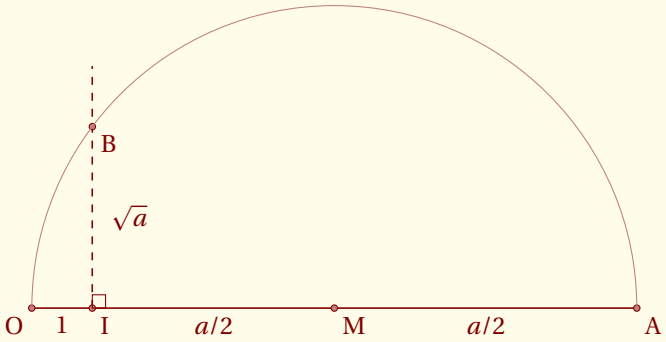
SECTION 4

Exemple minimal, mais complet

Cet exemple se trouve dans le dossier du package, et vous permet de tester votre installation.

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.

IM = a, OI = 1



Commentaires

Voyons tout d'abord le préambule. Il faut charger **xcolor.sty** avant **tkz-euclide.sty** c'est à dire avant **TikZ**. Les options de **xcolor.sty** dépendent des couleurs que vous utiliserez. Sinon, Il n'y rien de particulier à signaler, à l'exception du fait que **TikZ** peut poser des problèmes avec les caractères actifs de **frenchb** de **babel**, aussi j'ai créé deux macros **\tkzActivOff** et **\tkzActivOn** pour désactiver puis réactiver ces caractères.

```
\documentclass{scrartcl}
\usepackage[utf8]{inputenc}
\usepackage[upright]{fourier}
\usepackage[usenames,dvipsnames,svgnames]{xcolor}
\usepackage{tkz-euclide}
\usetkzobj{all} % on charge tous les objets
\usepackage[frenchb]{babel}
```

Commentaires

Le code suivant comprend quatre parties :
- la première prépare le support. Ici, les deux lignes **2** et **3** permettent de limiter la taille du dessin.
- la deuxième comprend les définitions de points nécessaires à la contruction, ce sont les lignes qui vont de **4** et **9** ;
- la troisième comprend les différents tracés, les lignes de 10 et 14 ;
- la dernière ne s'occupe que du placement des labels.

1. Mise en place

```
\begin{tikzpicture}[scale=.8]
   \tkzInit[ymin=-1,ymax=5,xmin=-1,xmax=10]
   \tkzClip

```

2. Création des points

```
   \tkzDefPoint(0,0){O}
   \tkzDefPoint(1,0){I}
   \tkzDefPointBy[homothety=center O ratio  10 ](I)  \tkzGetPoint{A}
   \tkzDefMidPoint(O,A)              \tkzGetPoint{M}
   \tkzDefPointWith[orthogonal](I,M) \tkzGetPoint{H}
   \tkzInterLC(I,H)(M,A)             \tkzGetSecondPoint{B}

```

3. Tracés

```
   \tkzDrawSegment(O,A)
   \tkzDrawSegment[style=dashed](I,H)
   \tkzDrawPoints(O,I,A,B,M)
   \tkzDrawArc(M,A)(O)
   \tkzMarkRightAngle(A,I,B)
```

4. Création des labels pour les points et les segments

```
   \tkzLabelSegment[right=4pt](I,B){$\sqrt{a}$}
   \tkzLabelSegment[below](O,I){$1$}
   \tkzLabelSegment[below](I,M){$a/2$}
   \tkzLabelSegment[below](M,A){$a/2$}
   \tkzLabelPoints(I,M,B,A)
   \tkzLabelPoint[below left](O){$O$}
\end{tikzpicture}
```

SECTION 5

Résumé de tkz-base

5.1 Utilité de tkz-base

tkz-base permet de simplifier l'utilisation d'intervalles de valeurs divers, ce package est nécessaire pour utiliser **tkz-tukey**, un package pour dessiner les représentations graphiques en statistiques élémentaires (ce package n'est pas encore en version officielle). Il est aussi nécessaire avec **tkz-fct**, pas plus officiel que le précédent et qui permet de dessiner les représentations graphiques des fonctions. Il utile également avec **tkz-euclide**, mais pas pour les mêmes raisons, car l'unité par défaut, le cm, convient parfaitement.

Premièrement, il faut savoir qu'il n'est pas nécessaire de s'occuper avec **TikZ** de la taille du support (background). Cependant il est parfois nécessaire, soit de tracer une grille, soit de tracer des axes, soit de travailler avec une unité différente que le centimètre, soit finalement 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**. Par exemple, si l'on veut travailler sur un carré de 10 cm de côté, mais tel que l'unité soit le dm alors il faudra utiliser.

```
\tkzInit[xmax=1,ymax=1,xstep=0.1,ystep=0.1]
```

en revanche pour des valeurs de x comprises entre 0 et 10 000 et des valeurs de y comprises entre 0 et 100 000, il faudra écrire

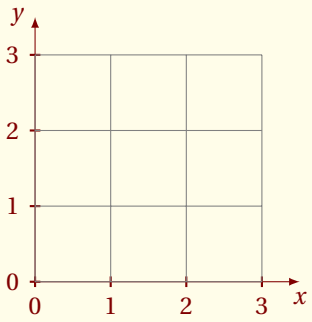
```
\tkzInit[xmax=10000,ymax=100000,xstep=1000,ystep=10000]
```

Tout cela a peu de sens pour faire de la géométrie euclidienne, et dans ce cas, il est recommandé de laisser l'unité graphique égale à 1 cm. Je n'ai d'ailleurs pas testé si toutes les macros destinées à la géométrie euclidienne, acceptaient d'autres valeurs que **xstep=1** et **ystep=1**. En revanche pour certains dessins, il est intéressant de fixer les valeurs extrêmes et de « clipper » le rectangle de définition afin de contrôler au mieux la taille de la figure.

Les principales macros de **tkz-base** sont :
- **\tkzInit**
- **\tkzClip**
- **\tkzAxeXY**
- **\tkzAxeX**
- **\tkzAxeY**
- **\tkzDrawX**
- **\tkzDrawY**
- **\tkzLabelX**
- **\tkzLabelY**
- **\tkzGrid**
- **\tkzRep**

Vous trouverez de multiples exemples dans la documentation de **tkz-base**.

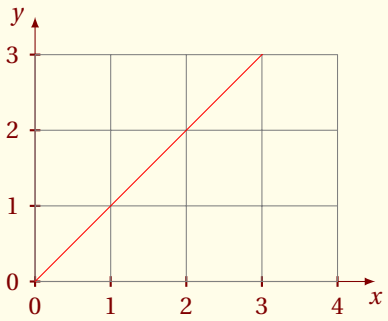
5.2 Exemple avec \tkzInit




```
\begin{tikzpicture}
 \tkzInit[xmax=3,ymax=3]
 \tkzAxeXY
 \tkzGrid
\end{tikzpicture}
```

5.3 \tkzClip

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

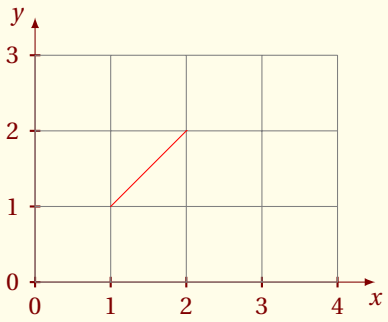



```
\begin{tikzpicture}
 \tkzInit[xmax=4, ymax=3]
 \tkzAxeXY
 \tkzGrid
 \tkzClip
 \draw[red] (-1,-1)--(5,5);
\end{tikzpicture}
```

Il est possible d'ajouter un peu d'espace

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

5.4 \tkzClip et l'option space

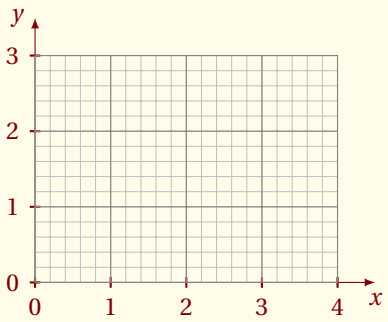



```
\begin{tikzpicture}
 \tkzInit[xmax=4, ymax=3]
 \tkzAxeXY
 \tkzGrid
 \tkzClip[space=-1]
 \draw[red] (-1,-1)--(5,5);
\end{tikzpicture}
```

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

5.5 \tkzGrid et l'option sub

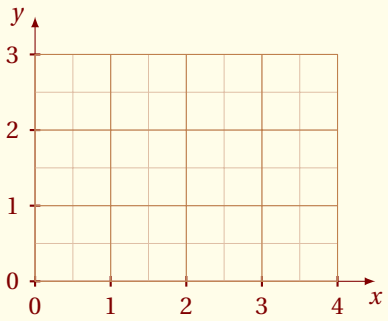
L'option **sub** permet d'afficher un grille secondaire plus fine.




```
\begin{tikzpicture}
 \tkzInit[xmax=4, ymax=3]
 \tkzAxeXY
 \tkzGrid[sub]
\end{tikzpicture}
```

5.6 \tkzGrid et les couleurs

L'option **sub** permet d'afficher un grille secondaire plus fine.




```
\begin{tikzpicture}
 \tkzInit[xmax=4, ymax=3]
 \tkzAxeXY
 \tkzGrid[sub,color=bistre,
          subxstep=.5,subystep=.5]
\end{tikzpicture}
```

SECTION 6

Les points

J'ai fait une distinction entre le point utilisé en géométrie euclidienne et le point pour représenter un élément d'un nuage statistique. Dans le premier cas, j'utilise comme objet un **node**, ce qui se traduit par le fait que la représentation du point ne peut être modifiée par un **scale** ; dans le second cas, j'utilise comme objet un **plot mark**. Ce dernier peut être mis à l'échelle et posséder des formes plus variées que le node.

La nouvelle macro est **\tkzDefPoint**, celle-ci permet d'utiliser des options propres à **TikZ** comme shift et les valeurs sont traitées avec tkz-base. De plus, si des calculs sont nécessaires alors c'est le package **fp.sty** qui s'en charge. On peut utiliser les coordonnées cartésiennes ou polaires.

6.1 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:r	no default	a est un angle en degré, r une dimension

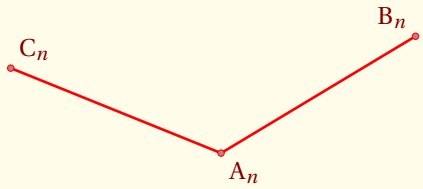
Les arguments obligatoires de cette macro sont deux dimensions exprimées avec des décimaux, dans le premier cas ce sont deux mesures de longueur, dans le second ce sont une mesure de longueur et la mesure d'un angle en degré

options	défaut	définition
shift	(0,0)	espacement entre deux valeurs
label	no default	permet de placer un label à une distance prédéfinie

*Toutes les options de **TikZ** que l'on peut appliquer à **coordinate**, sont applicables (enfin je l'espère !)*

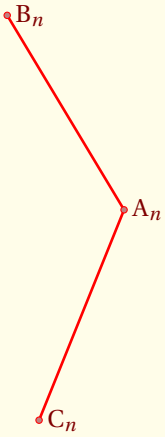
6.1.1 Utilisation de shift et label

shift permet de placer les points par rapport à un autre. Je n'aime guère utiliser l'option **label** mais en tout cas c'est possible. Attention à l'utilisation de **shift**, dans certains comme celui ci-dessous, une transformation générale de la figure n'est pas possible. Voir la méthode




```
\begin{tikzpicture}
 \tkzDefPoint[label=-60:$A_n$](2,3){A}
 \tkzDefPoint[shift={(2,3)},%
     label=above left:$B_n$](31:3){B}
 \tkzDefPoint[shift={(2,3)},%
     label=above right:$C_n$](158:3){C}
 \tkzDrawSegments[color=red,%
          line width=1pt](A,B A,C)
 \tkzDrawPoints[color=red](A,B,C)
\end{tikzpicture}
```

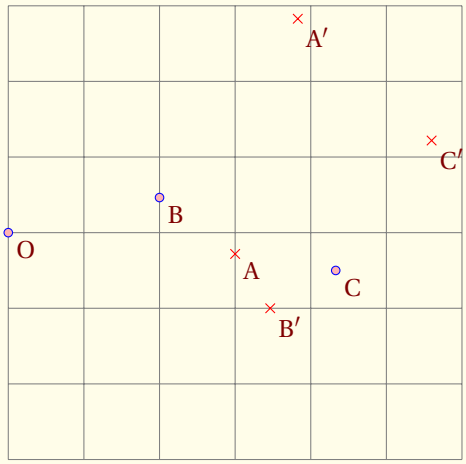
Préférable pour effectuer une rotation, est d'utiliser un environnement **scope**.




```
\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}
```

6.1.2 Formules et coordonnées

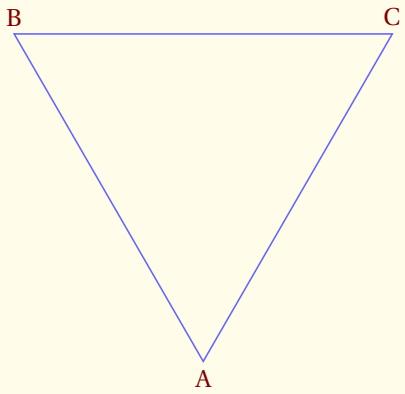
Il faut ici respecter la syntaxe de **fp.sty**. Il est toujours possible de passer par **pgfmath.sty** mais dans ce cas, il faut calculer les coordonnées avant d'utiliser la macro **\tkzDefPoint**.




```
\begin{tikzpicture}[scale=1]
  \tkzInit[xmax=6,ymax=6]
  \tkzGrid
  \tkzSetUpPoint[shape = circle,color = red,%
                 size = 8,fill = red!30]
  \tkzDefPoint(-1+1,-1+4){O}
  \tkzDefPoint({3*ln(exp(1))},{exp(1)}){A}
  \tkzDefPoint({4*sin(FPpi/6)},{4*cos(FPpi/6)}){B}
  \tkzDefPoint({4*sin(FPpi/3)},{4*cos(FPpi/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,%
                 shape=cross out](B',A,A',C')
  \tkzLabelPoints(A,O,B,B',A',C,C')
\end{tikzpicture}
```

6.1.3 Scope et \tkzDefPoint

On peut tout d'abord utiliser l'environnement **scope** de **TikZ** Dans l'exemple suivant, nous avons un moyen de définir un triangle isocèle.



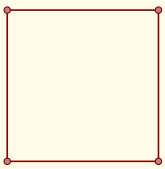

```
\begin{tikzpicture}[scale=1]
  \tkzSetUpLine[color=blue!60]
 \begin{scope}[rotate=30]
  \tkzDefPoint(2,3){A}
  \begin{scope}[shift=(A)]
     \tkzDefPoint(90:5){B}
     \tkzDefPoint(30:5){C}
  \end{scope}
 \end{scope}
 \tkzDrawPolygon(A,B,C)
\tkzLabelPoints[above](B,C)
\tkzLabelPoints[below](A)
\end{tikzpicture}
```

6.2 Définition de points multiples en coordonnées cartésiennes : \tkzDefPoints

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

x_1 et y_1 sont les coordonnées d'un point référencé n_1

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



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

6.3 Point relativement à un autre : \tkzDefShiftPoint

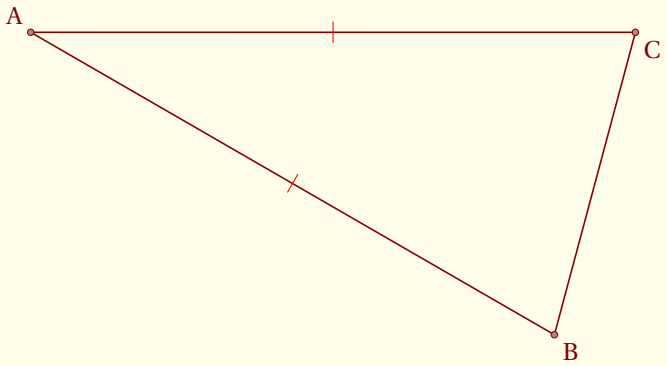
\tkzDefShiftPoint[⟨*Point*⟩](⟨*x,y*⟩){⟨*name*⟩} ou (⟨*a:r*⟩){⟨*name*⟩}

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
point	no default	**\tkzDefShiftPoint**[A](0:4){B}

Pas d'option. Le nom du point est obligatoire.

6.3.1 Exemple 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}[scale=2,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}
```

6.4 Point relativement à un autre : \tkzDefShiftPointCoord

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

Il s'agit d'effectuer une translation de vecteur (a,b) au point défini par rapport à l'oigine.

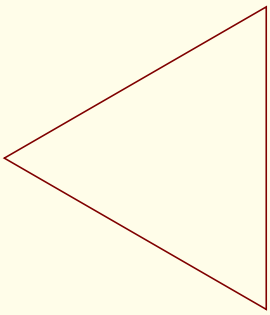
arguments	défaut	définition
(x,y)	no default	x et y sont deux dimensions, par défaut en cm.
(a:r)	no default	a est un angle en degré, r une dimension

options	défaut	exemple
a,b	no default	**\tkzDefShiftPointCoord**[2,3](0:4){B}

L'option est obligatoire

6.4.1 Triangle équilatéral avec \tkzDefShiftPointCoord

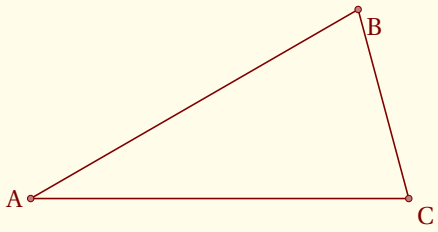
Voyons comment obtenir un triangle équilatéral (il y a beaucoup plus simple)



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

6.4.2 Triangle isocèle avec \tkzDefShiftPointCoord

Voyons comment obtenir un triangle isocèle dont l'angle principal est de 30 degrés. La rotation est possible. AB = AC = 5 et $\widehat{BAC}$




```
\begin{tikzpicture}[rotate=15]
 \tkzDefPoint(2,3){A}
 \tkzDefShiftPointCoord[2,3](15:5){B}
 \tkzDefShiftPointCoord[2,3](-15:5){C}
 \tkzDrawSegments(A,B B,C C,A)
 \tkzDrawPoints(A,B,C)
 \tkzLabelPoints(B,C)
 \tkzLabelPoint[left](A){$A$}
\end{tikzpicture}
```

6.5 Tracer des points \tkzDrawPoint

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

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

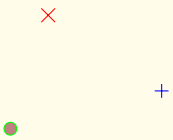
L'argument est obligatoire. Le disque prend la couleur du cercle mais 50% plus clair. Il est possible de tout modifier. Le point est un node et donc il est invariant si le dessin est modifié par une mise à l'échelle.

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***

6.5.1 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[shape=cross out,size=12,color=red](A)
 \tkzDrawPoint[shape=cross,size=12,color=blue](B)
 \tkzDrawPoint[size=12,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.

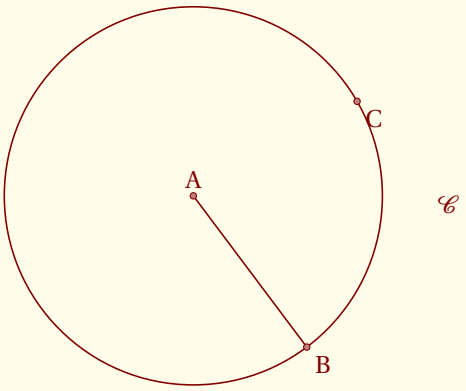
\tkzDrawPoints[⟨*local options*⟩](⟨*liste*⟩)

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.

6.5.2 Exemple avec \tkzDefPoint et \tkzDrawPoints


```
\begin{tikzpicture}[scale=.5]
 \tkzDefPoint(1,3){A}
 \tkzDefPoint(4,1){B}
 \tkzDefPoint(0,0){O}
 \tkzDrawPoints[size=8,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}
```

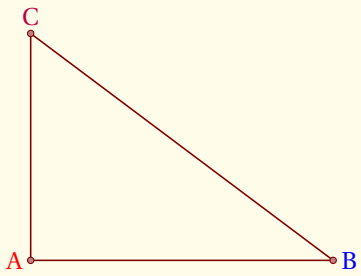
6.6 Ajouter des labels aux points \tkzLabelPoint

\tkzLabelPoint[⟨*local options*⟩](⟨*point*⟩){⟨*label*⟩}

arguments	exemple
point	**\tkzLabelPoint(A){A_1}**

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

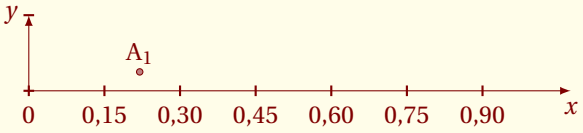
6.6.1 Exemple avec \tkzLabelPoint




```
\begin{tikzpicture}
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(4,0){B}
  \tkzDefPoint(0,3){C}
  \tkzDrawSegments(A,B B,C C,A)
  % \tkzDrawPolygon with
  % \usetkzobj{polygons}
  \tkzDrawPoints(A,B,C)
  \tkzLabelPoint[left,red](A){$A$}
  \tkzLabelPoint[right,blue](B){$B$}
  \tkzLabelPoint[above,purple](C){$C$}
\end{tikzpicture}
```

6.6.2 label et référence

La référence d'un point est l'objet qui permet d'utiliser le point, le label est le nom du point qui sera affiché.




```
\begin{tikzpicture}
   \tkzInit[xmax=1,xstep=0.15,ymax=.5]
   \tkzAxeX \tkzDrawY
   \tkzDefPoint(0.22,0.25){A}
   \tkzDrawPoint(A)
   \tkzLabelPoint[above](A){$A_1$}
 \end{tikzpicture}
```

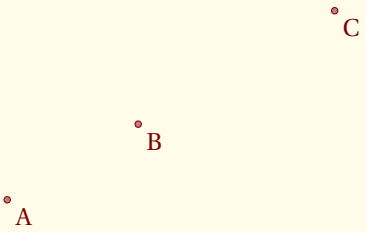
Il est possible de placer plusieurs labels rapidement quand les références des points sont identiques aux labels et quand les labels sont placés de la même manière par rapport aux points. Par défaut, c'est **below right** qui a été choisi.

\tkzLabelPoints[⟨*local options*⟩](⟨A1,A2,...⟩)

arguments	exemple	résultat
list of points	**\tkzLabelPoint(A,B,C)**	Affichage de A, B et C

Cette macro diminue le nombre de lignes de codes mais il n'est pas évident que tous les points aient besoin du même positionnement des labels.

6.6.3 Exemple avec \tkzLabelPoints



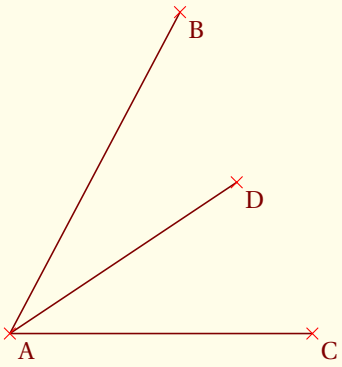

```
\begin{tikzpicture}
  \tkzDefPoint(2,3){A}
  \tkzDefShiftPoint[A](30:2){B}
  \tkzDefShiftPoint[A](30:5){C}
  \tkzDrawPoints(A,B,C)
  \tkzLabelPoints(A,B,C)
\end{tikzpicture}
```

6.7 Style des points avec \tkzSetUpPoint

\tkzSetUpPoint[⟨*local options*⟩]

options	défaut	définition
liste	no default	exemple **\tkzLabelPoint(A,B,C)**

Il s'agit d'une macro permettant de choisir un style pour les points. La macro **\tkzDrawSegments** est décrite ici.




```
\begin{tikzpicture}
  \tkzInit[ymin=-0.5,ymax=3,xmin=-0.5,xmax=7]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(02.25,04.25){B}
  \tkzDefPoint(4,0){C}
  \tkzDefPoint(3,2){D}
  \tkzDrawSegments(A,B A,C A,D)
  \tkzSetUpPoint[shape=cross out,size=10,color=red]
  \tkzDrawPoints(A,B,C,D)
  \tkzLabelPoints(A,B,C,D)
\end{tikzpicture}
```

SECTION 7

Points particuliers

L'introduction des points a été réalisée dans **tkz-base**. La macro la plus importante étant **\tkzDefPoint**. **\tkzDrawPoint** permet de tracer les points, quant à **\tkzLabelPoint**, elle permet d'afficher un label, lié au point. Voici quelques points particuliers.

7.1 Milieu d'un segment \tkzDefMidPoint

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

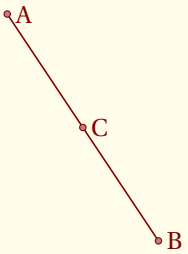
\tkzDefMidPoint(⟨*pt1,pt2*⟩)

Le résultat est dans **tkzPointResult**. On peut le récupérer avec **\tkzGetPoint**. Soit vous ne voulez pas conserver ce point et dans ce cas, vous pouvez immédiatement travailler avec **tkzPointResult**, soit vous aurez besoin untéreurement

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

7.1.1 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}
```

7.2 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. α_1, α_2, …, α_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=nb1,pt2=nb2,...*⟩)

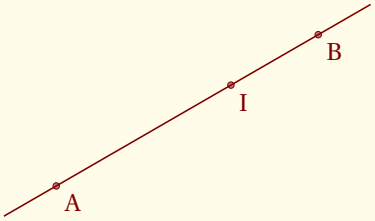
arguments	défaut	définition
(pt1=α_1,pt2=α_2,...)	no default	Chaque point a une pondération

Il faut au moins deux points.

7.2.1 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 :

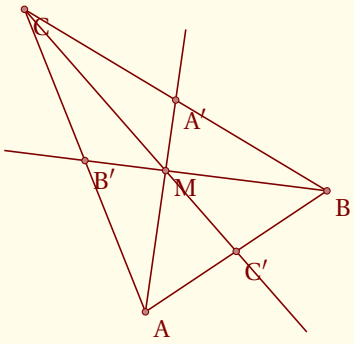
$$\overrightarrow{AI} = \frac{2}{3}\overrightarrow{AB}$$




```
\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}
```

7.2.2 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]
 \tkzInit[xmax=6,ymax=6]
 \tkzDefPoint(2,1){A}
 \tkzDefPoint(5,3){B}
 \tkzDefPoint(0,6){C}
 \tkzDrawPolygon(A,B,C)
 \tkzDefBarycentricPoint(A=1,B=1,C=1)
 \tkzGetPoint{M}
 \tkzDrawLines[add=0 and 1](A,M B,M C,M)
 \tkzDrawPoints(A,B,C,M)
 \tkzLabelPoints(A,B,C,M)
 \tkzDefMidPoint(A,B)  \tkzGetPoint{C'}
 \tkzDefMidPoint(A,C)  \tkzGetPoint{B'}
 \tkzDefMidPoint(C,B)  \tkzGetPoint{A'}
 \tkzDrawPoints(A',B',C')
 \tkzLabelPoints(A',B',C')
\end{tikzpicture}
```

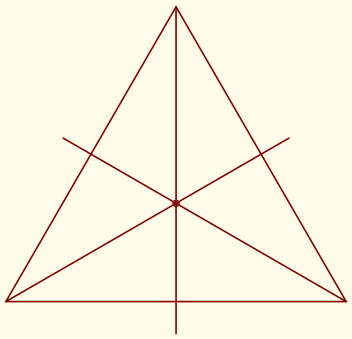
7.3 \tkzCentroid

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

\tkzCentroid(⟨*pt1,pt2,pt3*⟩)

arguments	défaut	définition
(pt1,pt2,pt3)	no default	liste non ordonnée de trois points

7.3.1 Utilisation de \tkzCentroid



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

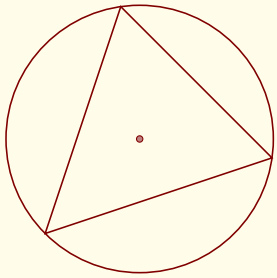
7.4 \tkzCircumCenter

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

\tkzCircumCenter(⟨*pt1,pt2,pt3*⟩)

arguments	défaut	définition
(pt1,pt2,pt3)	no default	liste non ordonnée de trois points

7.4.1 Utilisation de \tkzCircumCenter



```
\begin{tikzpicture}
 \tkzDefPoint(0,1){A} \tkzDefPoint(3,2){B}
 \tkzDefPoint(1,4){C}
 \tkzDrawPolygon[color=Maroon](A,B,C)
 \tkzCircumCenter(A,B,C)\tkzGetPoint{G}
 \tkzDrawPoint(G)
 \tkzDrawCircle(G,A)
\end{tikzpicture}
```

7.5 \tkzInCenter

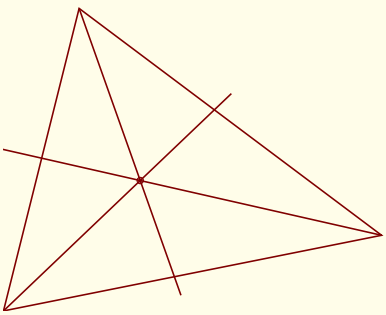
On obtient le centre du cercle inscrit du triangle. Le résultat est bien sûr dans **tkzPointResult**. On peut le récupérer avec **\tkzGetPoint**.

\tkzInCenter(⟨*pt1,pt2,pt3*⟩)

arguments	défaut	définition
(pt1,pt2,pt3)	no default	liste non ordonnée de trois points

7.5.1 Utilisation de \tkzInCenter avec trois points

Les trois points sont donnés dans le sens direct



```
\begin{tikzpicture}
  \tkzInit[xmax=6,ymax=6]
  \tkzClip
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(5,1){B}
  \tkzDefPoint(1,4){C}
  \tkzDrawPolygon[color=Maroon](A,B,C)
  \tkzInCenter(A,B,C)\tkzGetPoint{G}
  \tkzDrawPoint(G)
  \tkzDrawLines[add = 0 and 2/3](A,G B,G C,G)
\end{tikzpicture}
```

SECTION 8

Définition aléatoire de points

Il y a pour le moment quatre possibilités :

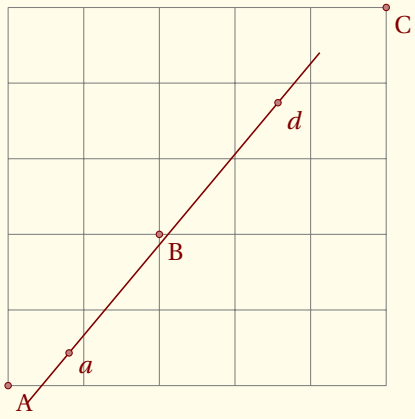
1. point dans un rectangle,
2. sur un segment,
3. sur une droite,
4. sur un cercle.

\tkzGetRandPointOn[⟨*local options*⟩]{⟨*name*⟩}

options	définition
rectangle = #1 and #2	#1 et #2 sont des noms de points
segment = #1--#2	#1 et #2 sont des noms de points
line = #1--#2	#1 et #2 sont des noms de points
circle = center #1 radius #1	#1 est un point et #1 une mesure

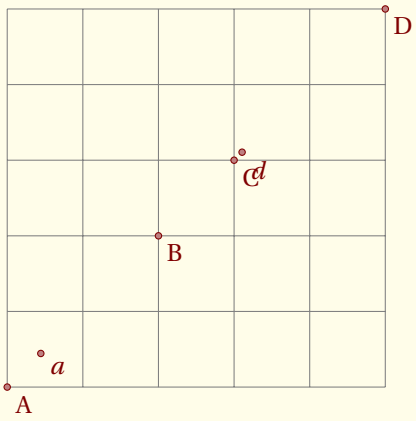
Cette macro est assez simple à utiliser, voyez les exemples.

8.1 Point aléatoire dans un rectangle



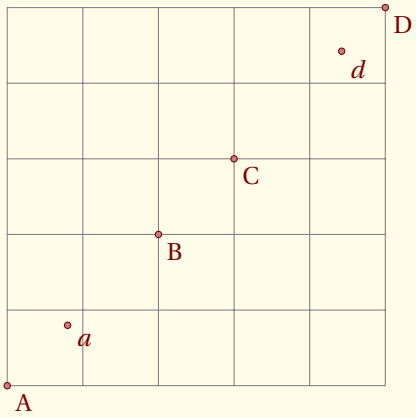

```
\begin{tikzpicture}
  \tkzInit[xmax=5,ymax=5]  \tkzGrid
  \tkzDefPoint(0,0){A}  \tkzDefPoint(2,2){B}
  \tkzDefPoint(5,5){C}
  \tkzGetRandPointOn[rectangle = A and B]{a}
  \tkzGetRandPointOn[rectangle = B and C]{d}
  \tkzDrawLine(a,d)
  \tkzDrawPoints(A,B,C,a,d)
  \tkzLabelPoints(A,B,C,a,d)
\end{tikzpicture}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

8.2 Point aléatoire sur un segment



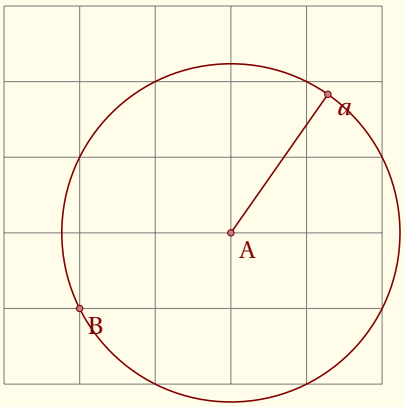

```
\begin{tikzpicture}
  \tkzInit[xmax=5,ymax=5] \tkzGrid
  \tkzDefPoint(0,0){A} \tkzDefPoint(2,2){B}
  \tkzDefPoint(3,3){C} \tkzDefPoint(5,5){D}
  \tkzGetRandPointOn[segment = A--B]{a}
  \tkzGetRandPointOn[segment = C--D]{d}
  \tkzDrawPoints(A,B,C,D,a,d)
  \tkzLabelPoints(A,B,C,D,a,d)
\end{tikzpicture}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

8.3 Point aléatoire sur une droite




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

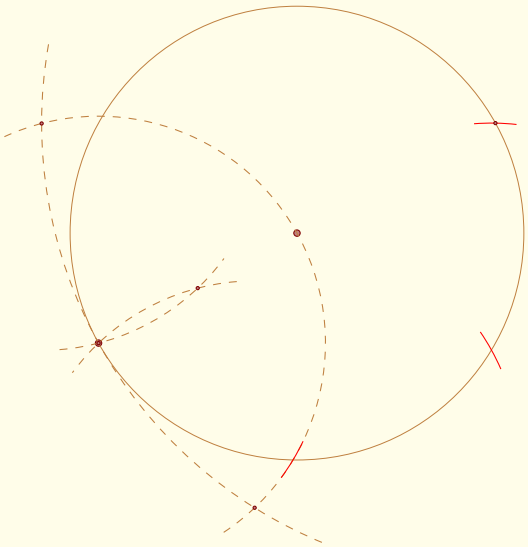
8.4 Point aléatoire sur un cercle




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

8.5 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}
  \tkzGetRandPointOn[circle= center A radius 4cm]{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)
 \end{tikzpicture}
```

SECTION 9

Définition de points par transformation ; \tkzDefPointBy

Ces transformations sont au nombre de sept :

1. la translation ;
2. l'homothetie ;
3. la réflexion ou symétrie orthogonale ;
4. la symétrie centrale ;
5. la projection orthogonale ;
6. la rotation ;
7. la rotation en radian ;
8. 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**. Dans le second cas, 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. Par défaut l'image de A est A′. Par exemple, on écrira :

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

La variante sans (s), évite l'usage d'une boucle et d'un test et est donc plus efficace.

\tkzDefPointBy[⟨*local options*⟩](⟨*pt*⟩)

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

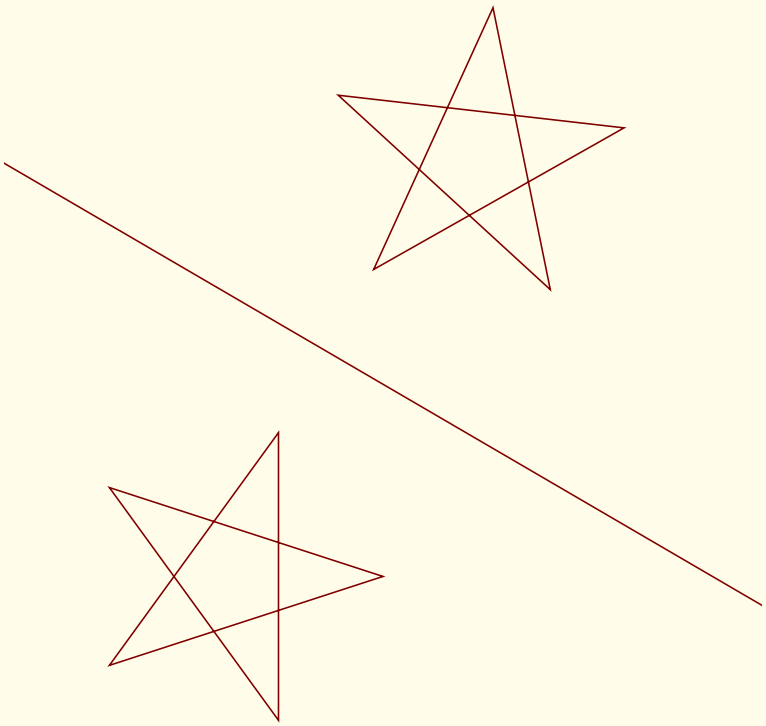
arguments	définition	exemples
pt	nom d'un point existant	(A)

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.

9.1 La réflexion ou symétrie orthogonale

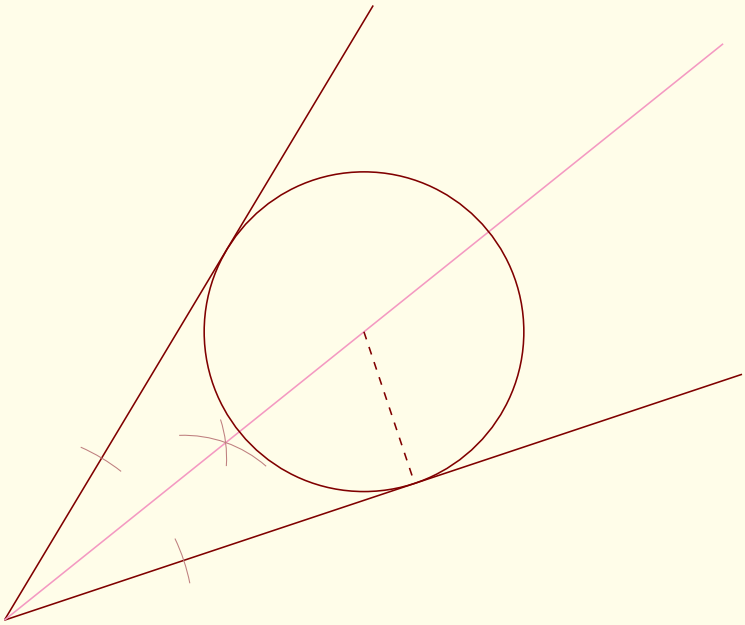
9.1.1 Exemple de réflexion



```
\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}
```

9.2 L'homothétie

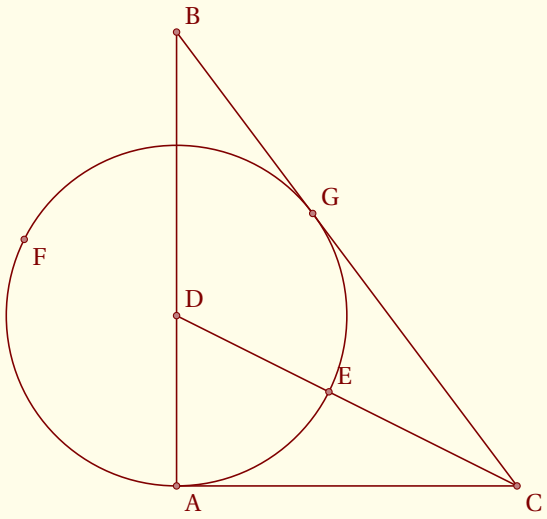
9.2.1 Exemple d'homothétie et de projection



```
\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[style=dashed](a',k)
  \tkzShowLine[bisector,size=2,gap=3](B,A,C)
  \tkzDrawCircle(a',k)
\end{tikzpicture}
```

9.3 La projection

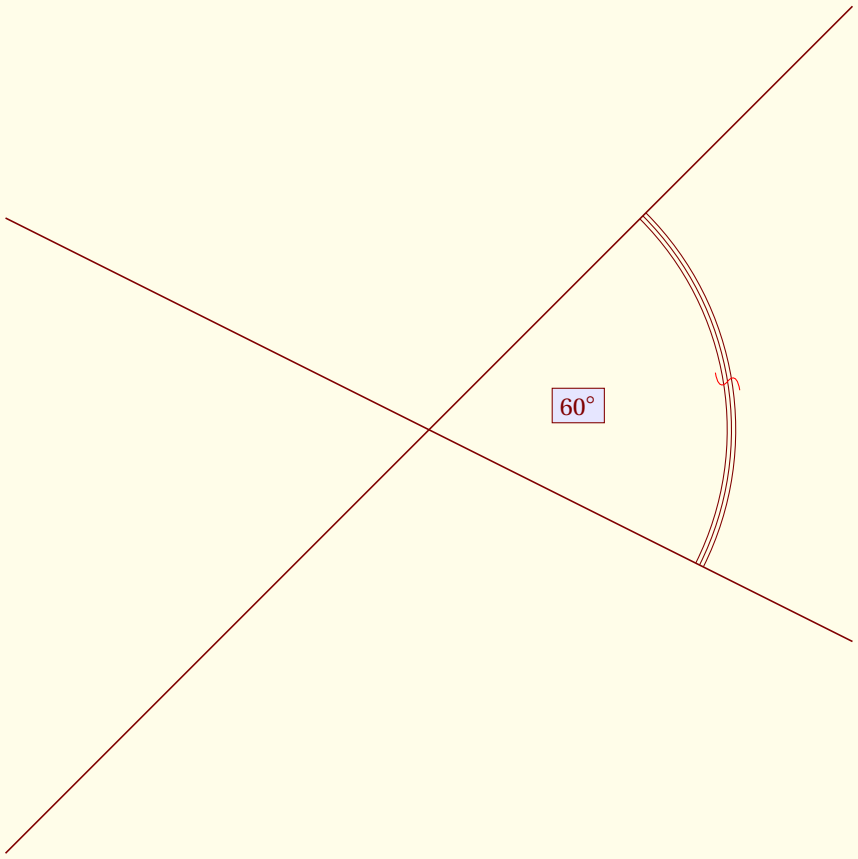
9.3.1 Exemple de projection




```
\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}
```

9.4 La symétrie

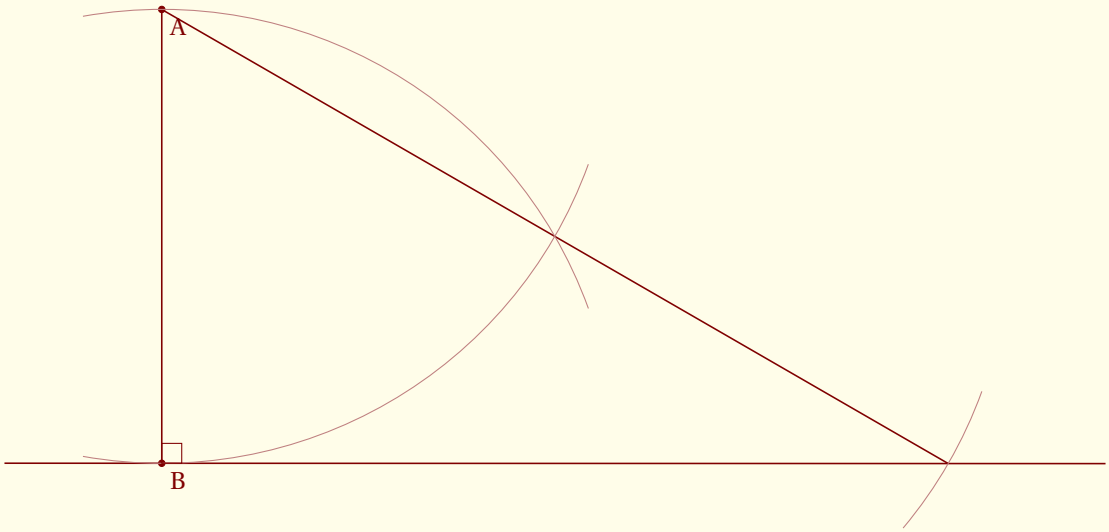
9.4.1 Exemple de symétrie




```
\begin{tikzpicture}[scale=2]
  \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){$60^{\circ}$}
\end{tikzpicture}
```

9.5 La rotation

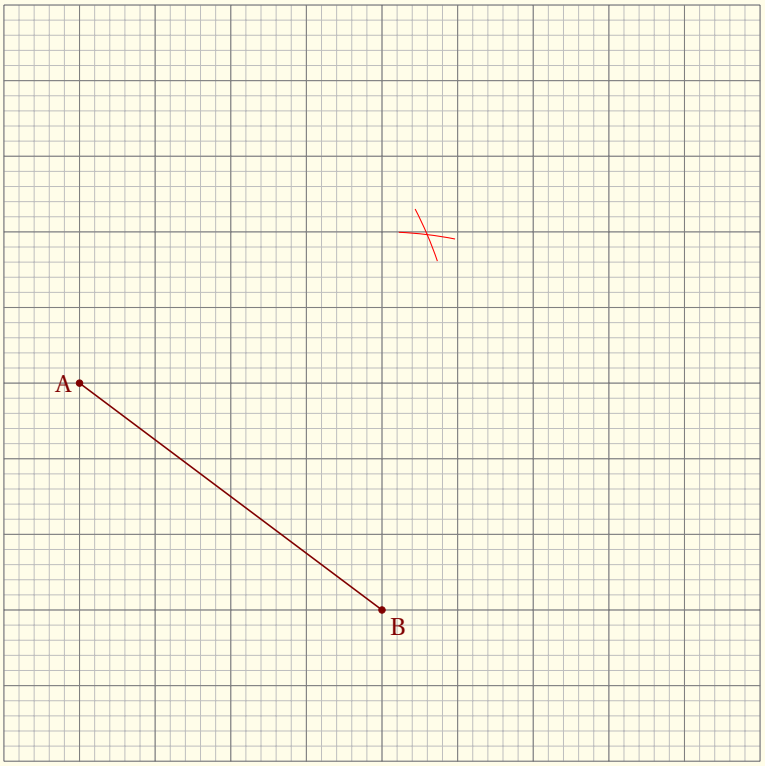
9.5.1 Exemple de rotation




```
\begin{tikzpicture}[scale=1.2,rotate=-90]
\tkzInit
\tkzPoint(0,0){A} \tkzPoint(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}
```

9.6 La rotation en radian

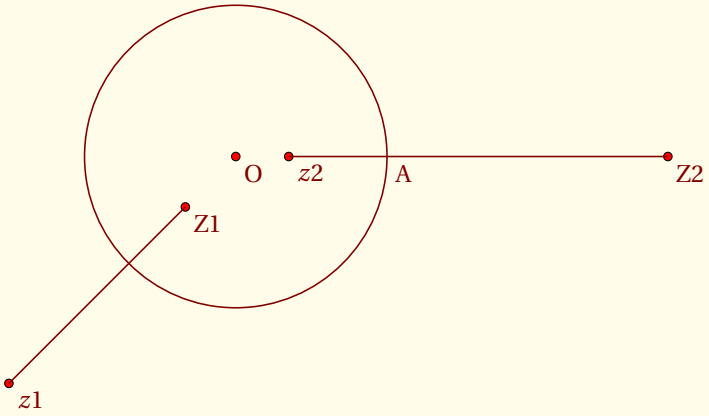
9.6.1 Exemple de rotation en radian




```
\begin{tikzpicture}
  \tkzInit\tkzGrid[sub]
  \tkzPoint[pos=left](1,5){A}
  \tkzPoint(5,2){B}
  \tkzDrawSegment(A,B)
  \tkzDefPointBy[rotation in rad= center A angle pi/3](B)
  \tkzGetPoint{C}
  \tkzCompass[color=red](A,C)
  \tkzCompass[color=red](B,C)
\end{tikzpicture}
```

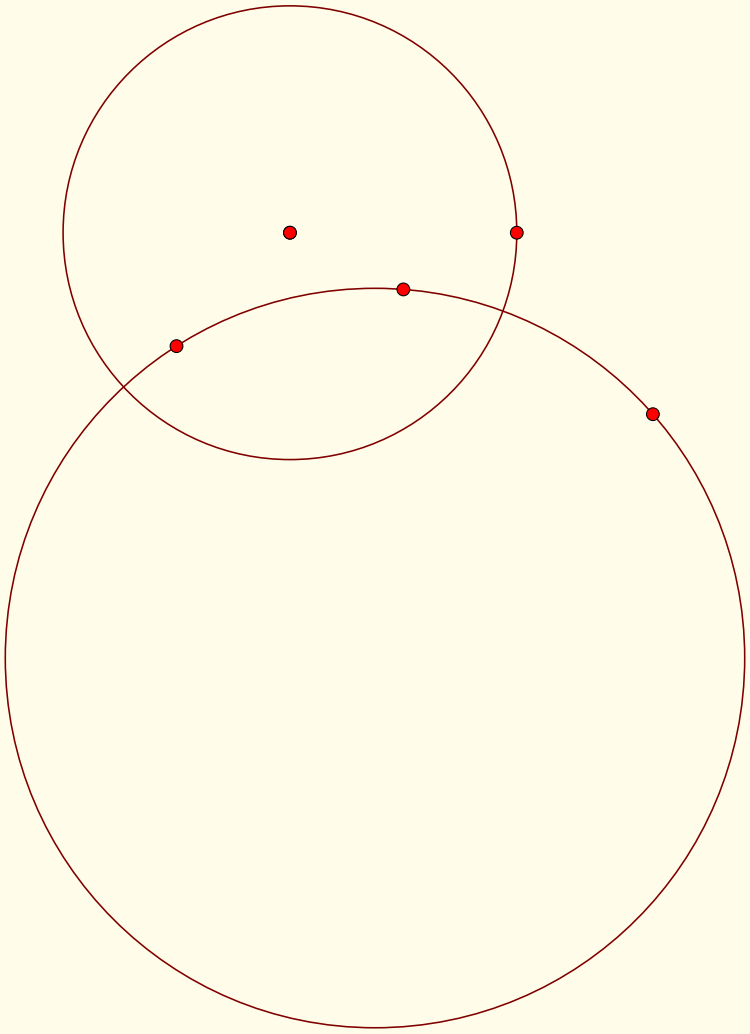
9.7 L'inversion par rapport à un cercle

9.7.1 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[fill=red,color=black,size=8](O,z1,z2)
  \tkzDefPointBy[inversion = center O through A](z1)
  \tkzGetPoint{Z1}
  \tkzDefPointBy[inversion = center O through A](z2)
  \tkzGetPoint{Z2}
  \tkzDrawPoints[fill=red,color=black,size=8](Z1,Z2)
  \tkzDrawSegments(z1,Z1 z2,Z2)
  \tkzLabelPoints(O,A,z1,z2,Z1,Z2)
\end{tikzpicture}
```

9.7.2 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=12](O,z1,z2,Z1,O,A)
\end{tikzpicture}
```

Il existe une variante de cette macro pour la définition de multiples images

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

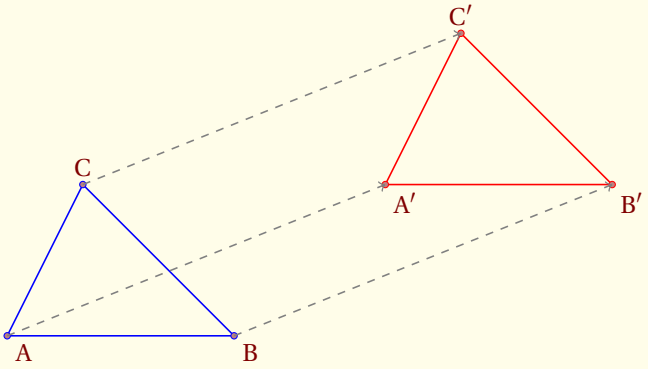
arguments	exemples
(⟨*liste de pts*⟩){⟨*liste de pts*⟩}	`(A,B){E,F}` E est l'image de A et F celle de B.

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

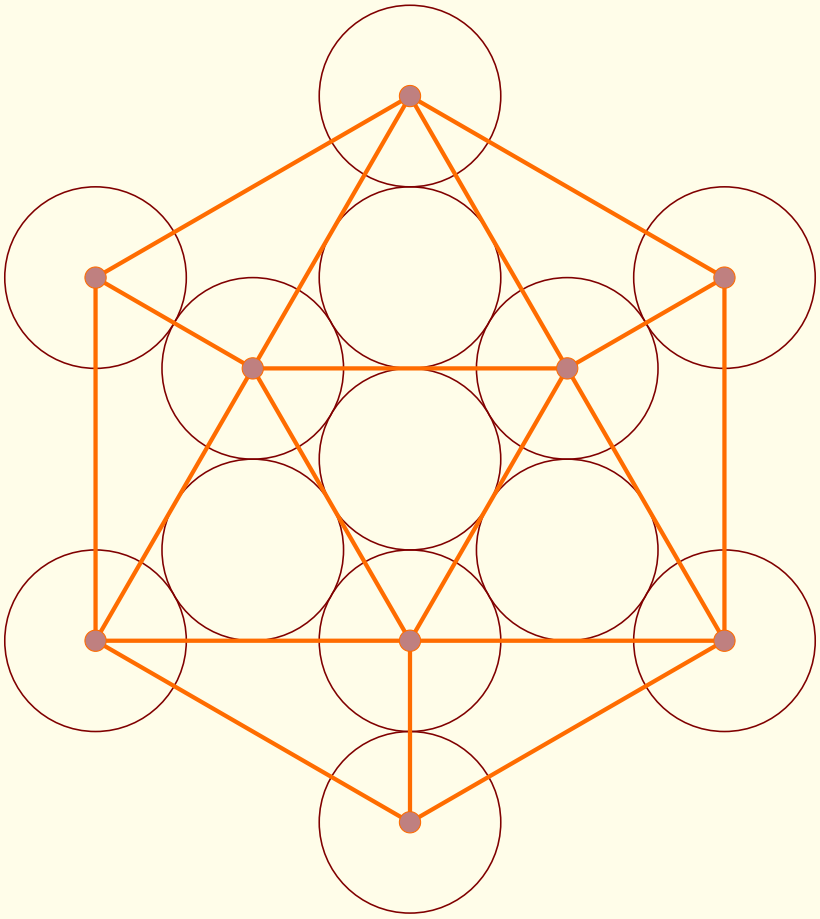
Les points sont seulement définis et non tracés.

9.8 Exemple de translation



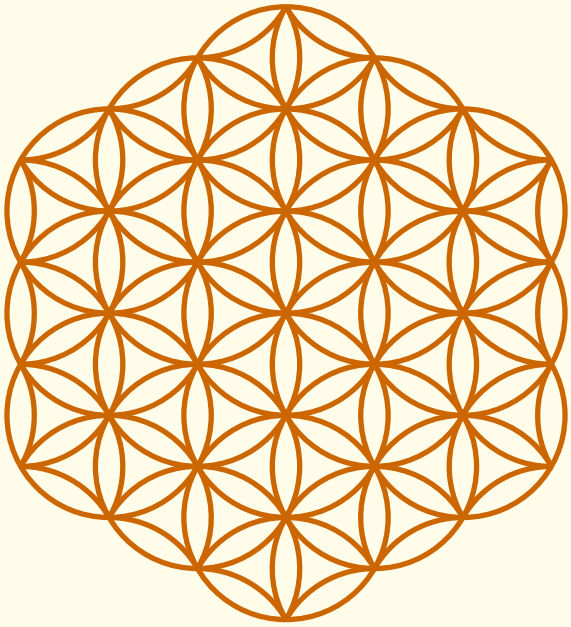

```
\begin{tikzpicture}
 \tkzDefPoint(0,0){A}  \tkzDefPoint(5,2){A'}
 \tkzDefPoint(3,0){B}  \tkzDefPoint(1,2){C}
 \tkzDefPointsBy[translation= from A to A'](B,C){}
 \tkzDrawPolygon[color=blue](A,B,C)
 \tkzDrawPolygon[color=red](A',B',C')
 \tkzDrawPoints[color=blue](A,B,C)
 \tkzDrawPoints[color=red](A',B',C')
 \tkzLabelPoints(A,B,A',B')  \tkzLabelPoints[above](C,C')
 \tkzDrawSegments[color = gray,->,style=dashed](A,A' B,B' C,C')
\end{tikzpicture}
```

9.9 Fruit of Life



```
\begin{tikzpicture}[scale=.8]
 \tkzDefPoint(0,0){O}  \tkzDefPoint(1.5,0){A}
 \tkzDrawCircle(O,A)
 \foreach \i in {0,...,5}{
  \tkzDefPointBy[rotation  = center O  angle 30+60*\i](A) \tkzGetPoint{a\i}
  \tkzDefPointBy[homothety = center O  ratio 2](a\i) \tkzGetPoint{b\i}
  \tkzDefPointBy[homothety = center O  ratio 3](a\i) \tkzGetPoint{c\i}
  \tkzDefPointBy[homothety = center O  ratio 4](a\i) \tkzGetPoint{d\i}
  \tkzDrawCircle(b\i,a\i) \tkzDrawCircle(d\i,c\i)
  }
\tkzDrawPolygon[color=red!50!Gold,ultra thick](d0,d1,d2,d3,d4,d5)
\tkzDrawPolygon[color=red!50!Gold,ultra thick](b0,b2,b4)
\tkzDrawSegments[color=red!50!Gold,ultra thick](b0,d5 b0,d0 b0,d1 %
                            b2,d1 b2,d2 b2,d3 b4,d3 b4,d4 b4,d5)
\tkzDrawPoints[color=red!50!Gold,size=20](b0,b2,b4,d0,d1,d2,d3,d4,d5)
\end{tikzpicture}
```

9.10 Flower of Life

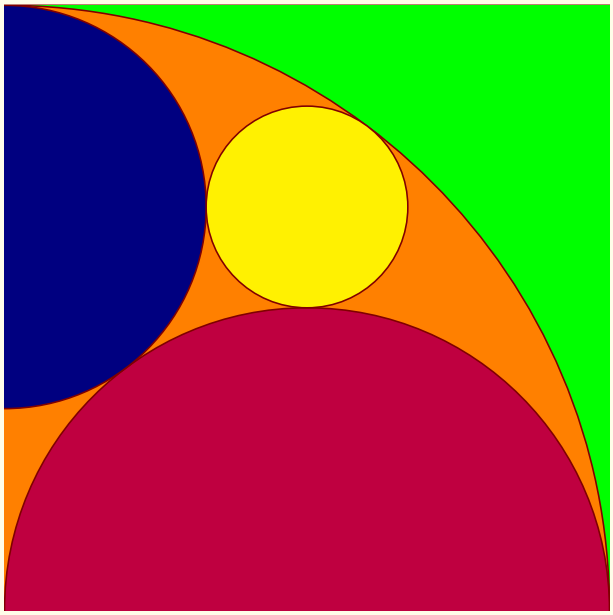


```
\begin{tikzpicture}[scale=.6]
 \tkzSetUpLine[line width=2pt,color=orange!80!black]
 \tkzSetUpCompass[line width=2pt,color=orange!80!black]
 \tkzDefPoint(0,0){O} \tkzDefPoint(2.25,0){A}
 \tkzDrawCircle(O,A)
 \foreach \i in {0,...,5}{
  \tkzDefPointBy[rotation= center O angle 30+60*\i](A)   \tkzGetPoint{a\i}
  \tkzDefPointBy[rotation= center {a\i} angle  120](O)   \tkzGetPoint{b\i}
  \tkzDefPointBy[rotation= center {a\i} angle  180](O)   \tkzGetPoint{c\i}
  \tkzDefPointBy[rotation= center {c\i} angle  120](a\i) \tkzGetPoint{d\i}
  \tkzDefPointBy[rotation= center {c\i} angle   60](d\i) \tkzGetPoint{f\i}
  \tkzDefPointBy[rotation= center {d\i} angle   60](b\i) \tkzGetPoint{e\i}
  \tkzDefPointBy[rotation= center {f\i} angle   60](d\i) \tkzGetPoint{g\i}
  \tkzDefPointBy[rotation= center {d\i} angle   60](e\i) \tkzGetPoint{h\i}
  \tkzDefPointBy[rotation= center {e\i} angle  180](b\i) \tkzGetPoint{k\i}

  \tkzDrawCircle(a\i,O) \tkzDrawCircle(b\i,a\i)
  \tkzDrawCircle(c\i,a\i)
  \tkzDrawArc[rotate](f\i,d\i)(-120)
  \tkzDrawArc[rotate](e\i,d\i)(180)
  \tkzDrawArc[rotate](d\i,f\i)(180)
  \tkzDrawArc[rotate](g\i,f\i)(60)
  \tkzDrawArc[rotate](h\i,d\i)(60)
  \tkzDrawArc[rotate](k\i,e\i)(60) }
 \tkzClipCircle(O,f0)
\end{tikzpicture}
```

9.11 Sangaku cercle et carré

Dans cet exemple, on peut voir comment utiliser un point sans le nommer



```
\begin{tikzpicture}[scale = 1]
   \tkzInit[xmax = 8] \tkzClip
   \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)
   \tkzTgtFromP(F,A)(B)
   \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}
```

9.12 Constructions de certaines transformations \tkzShowTransformation

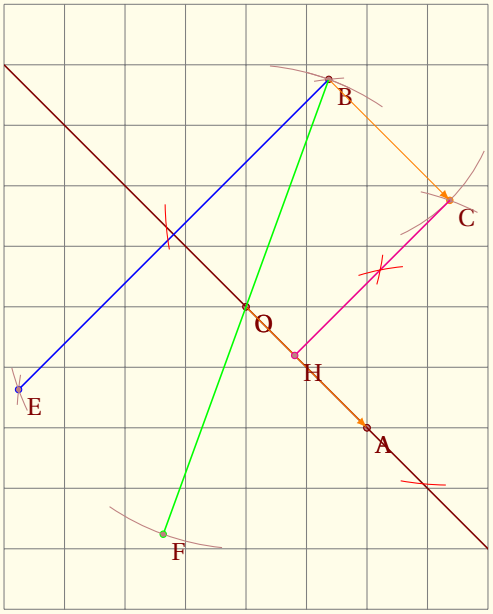
\tkzShowTransformation[⟨*local options*⟩](⟨*pt1,pt2*⟩) ou (⟨*pt1,pt2,pt3*⟩)

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)

Il faut ajouter bien sûr tous les styles de ***TikZ*** *pour les tracés*

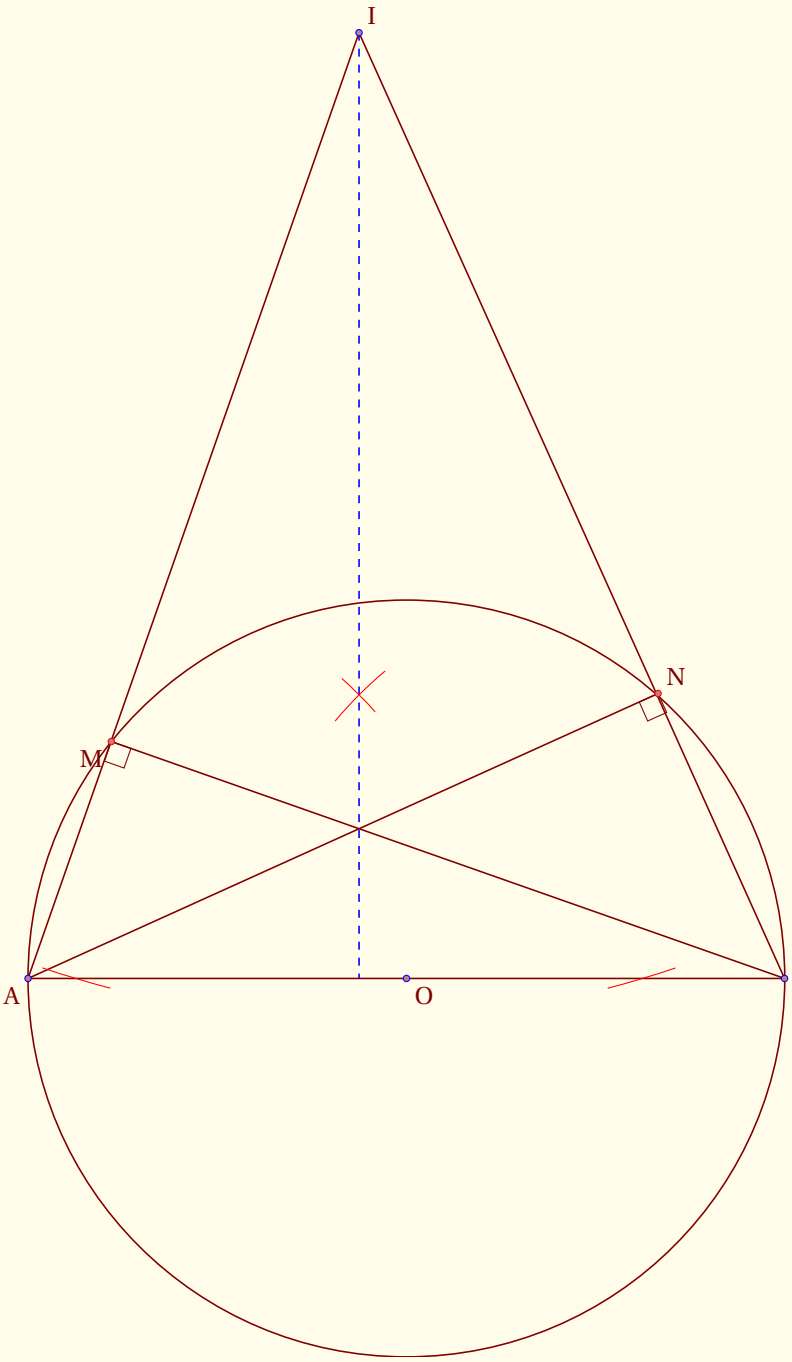
9.12.1 Exemple d'utilisation de \tkzShowTransformation




```
\begin{tikzpicture}[scale=.8]
  \tkzInit[xmin=-4,xmax=4,ymin=-5,ymax=5]
  \tkzGrid \tkzClip \tkzPoint(0,0){O} \tkzPoint(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)
  \tkzDrawVectors[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}
```

9.12.2 Autre exemple d'utilisation de \tkzShowTransformation

Vous retouverez cette figure, mais sans les traits de construction



```
  \begin{tikzpicture}[scale=1.25]
  % on définit les points nécessaires
  \tkzInit[ymin=-3]
  \tkzClip[space=1]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(8,0){B}
  \tkzDefPoint(3.5,10){I}
  \tkzDefMidPoint(A,B) \tkzGetPoint{O}
  % syntaxe (liste de points) {liste des images} si vide on met des '
  \tkzDefPointBy[projection=onto A--B](I) \tkzGetPoint{J}
  \tkzInterLC(I,A)(O,A) \tkzGetPoints{M'}{M}
  \tkzInterLC(I,B)(O,A)  \tkzGetPoints{N}{N'}
  \tkzDrawCircle[diameter](A,B)
   % attention plusieurs segments donc (s) espace entre les objets
   % virgule entre les points
  \tkzDrawSegments(I,A I,B A,B B,M A,N)
  % idem (s) et espace entre les objets
  \tkzMarkRightAngles(A,M,B A,N,B)
  \tkzDrawSegment[style=dashed,color=blue](I,J)
  % tkzShowTransformation il y a aussi tkzShowLine
  \tkzShowTransformation[projection=onto A--B,color=red,size=3,gap=-3](I)
  % on trace les points à la fin ainsi c'est plus propre, il n'y a rien
  % par-dessus
  \tkzDrawPoints[color=red](M,N)
  \tkzDrawPoints[color=blue](O,A,B,I)
  %  \tkzLabelPoints version rapide de  \tkzLabelPoint on met automatiquement
  % $O$ etc ... sinon on traite chaque point l'un après l'autre avec
  %  \tkzLabelPoint(le point){son label}
  \tkzLabelPoints(O)  \tkzLabelPoints[above right](N,I)
  \tkzLabelPoints[below left](M,A)
\end{tikzpicture}
```

SECTION 10

Intersections

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

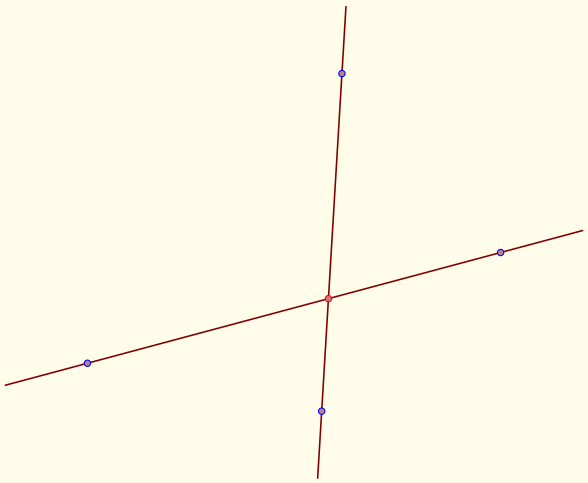
Les commandes associées n'ont pas d'arguments optionnels et l'usager doit lui même déterminer l'existence des points d'intersection.

10.1 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, son nom est placé entre accolades.*

10.1.1 exemple d'intersection entre deux droites



```
\begin{tikzpicture}[rotate=-30]
   \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}
```

De nombreux points particuliers sont obtenus avec cette macro par exemple l'orthocentre (OrthoCenter) voir **\tkzOrthoCenter**, le centre du cercle circonscrit à un triangle **\tkzCircumCenter**.

10.2 Intersection d'une droite et d'un cercle

Pour avoir une syntaxe homogène, l'option pour définir le cercle à l'aide de la mesure du rayon est **R** comme pour les macros pour le cercle , les arcs et les secteurs.

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(⟨A,B⟩**)(**⟨O,C/*r*⟩**){**⟨I⟩**}{**⟨J⟩**}**

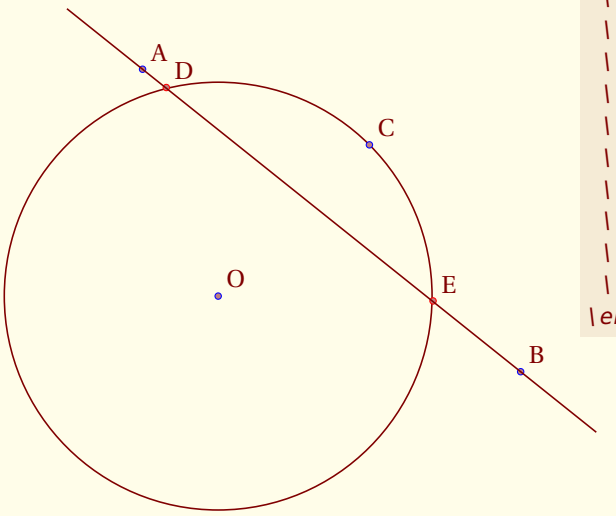
Les arguments sont donc deux couples. Le premier couple est un couple de points, le second est soit un couple de points si aucune option n'est utilisée ou bien si l'option **N** est utilisée sinon le couple est constitué d'un point (le centre du cercle et d'une mesure, celle du rayon).

options	défaut	définition
N	N	(O,C) détermine le cercle
R	N	(O, 1 cm) ou (O, 120 pt)

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*

10.2.1 Exemple simple d'intersection droite-cercle

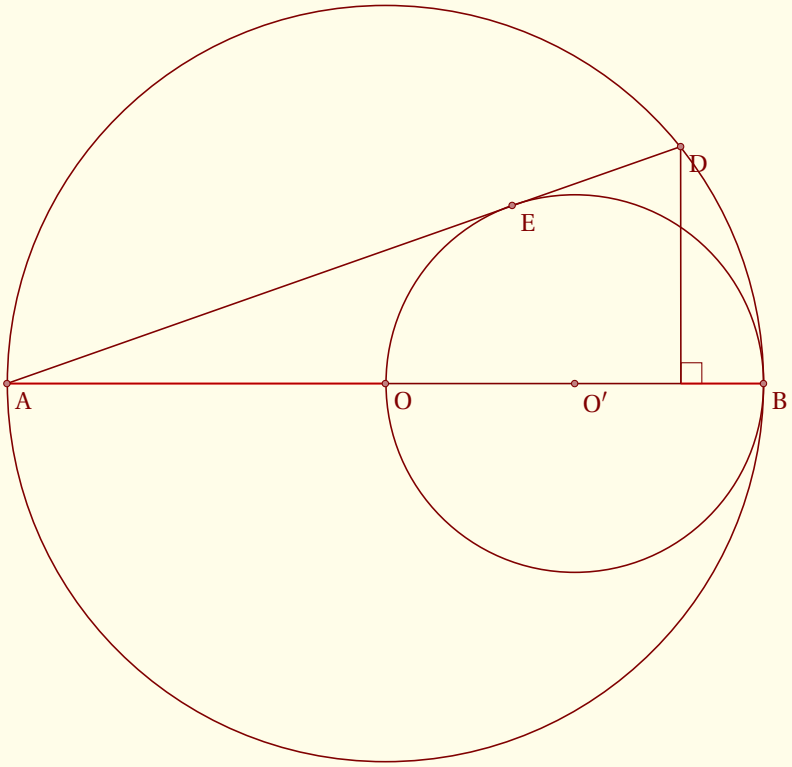
Dans l'exemple suivant, le tracé du cercle utilise deux points et l'intersection de la droite et du cercle utilise deux couples de points




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

10.2.2 Exemple plus complexe d'intersection droite-cercle

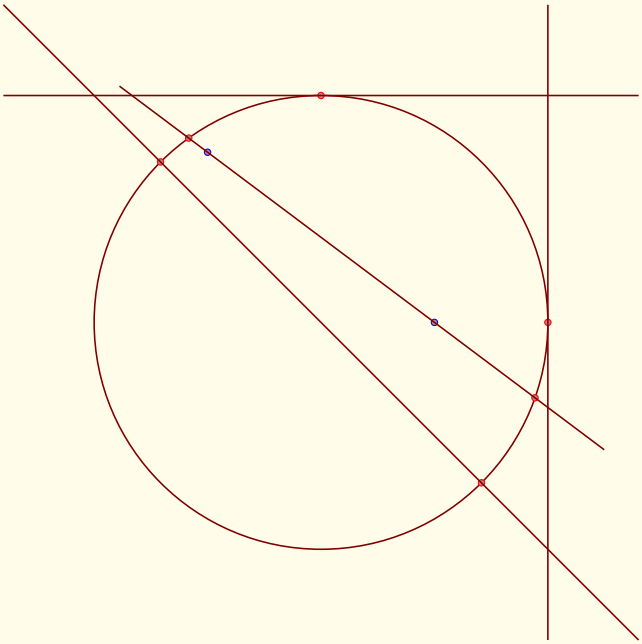
http://gogeometry.com/problem/p190_tangent_circle




```
\begin{tikzpicture}[scale=1.25]
  \tkzInit[xmin=0,xmax=8,ymin=-4,ymax=4]  \tkzClip[space=.4]
  \tkzDefPoint(0,0){A}  \tkzDefPoint(8,0){B}
  \tkzDefMidPoint(A,B)  \tkzGetPoint{O}
  \tkzDrawCircle(O,B)
  \tkzDefMidPoint(O,B)  \tkzGetPoint{O'}
  \tkzDrawCircle(O',B)
  \tkzTangent[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}
```

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

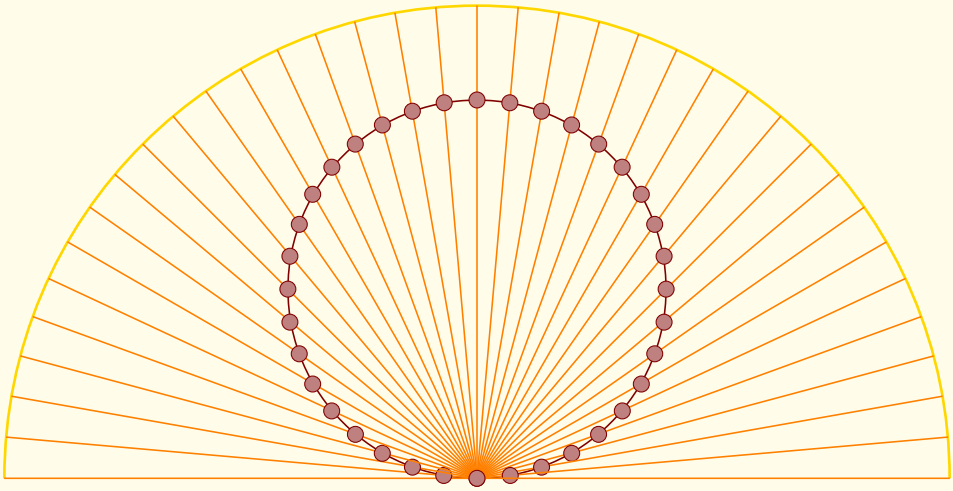
Regardons quelques cas particuliers comme des droites tangentes au cercle.



```
\begin{tikzpicture}[scale=.75]
  \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}
```

10.2.4 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 **fp**. Vous pouvez constater que j'utilise la macro **\FPpi** car **fp** travaille en radians !. De plus quand des calculs nécéssitent l'emploi de parenthèses, celles-ci doivent être insérées dans un groupe TEX{ ...}.

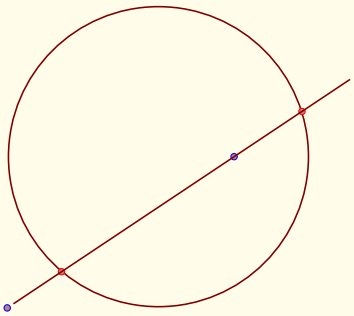


```
\begin{tikzpicture}[scale=2.5,rotate=180]
  \tkzDefPoint(0,1){J} \tkzDefPoint(0,0){O}
  \tkzDrawCircle[R](O,1 cm)
  \tkzDrawArc[R,line width=1pt,color=Gold](J,2.5 cm)(180,0)
  \foreach \i in {0,-5,-10,...,-85}{
     \tkzDefPoint({2.5*cos(\i*\FPpi/180)},{1+2.5*sin(\i*\FPpi/180)}){P}
     \tkzDrawSegment[color=orange](J,P)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

10.2.5 Calcul de la mesure du rayon

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

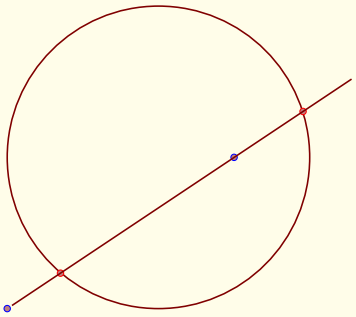
La mesure du rayon peut être le résultat d'un calcul que l'on ne fera pas au sein de la macro d'intersection, mais avant. On peut calculer une longueur de plusieurs façons. Il est possible bien sûr, d'utiliser le module **pgfmath** et la macro **\pgfmathsetmacro**. Dans certains, les résultats obtenus ne sont pas assez précis ainsi le calcul suivant $0.0002 \div 0.0001$ donne 1.98 avec pgfmath alors que fp.sty donnera 2. C'est pour cela que j'ai préféré interdire le calcul pendant le passage de paramètres, cela permet à chacun de choisir sa méthode.



```
\begin{tikzpicture}
  \tkzDefPoint(2,2){A}
  \tkzDefPoint(5,4){B}
  \tkzDefPoint(4,4){O}
  \pgfmathsetmacro{\tkzLen}{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}
```

10.2.6 Calcul de la mesure du rayon

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

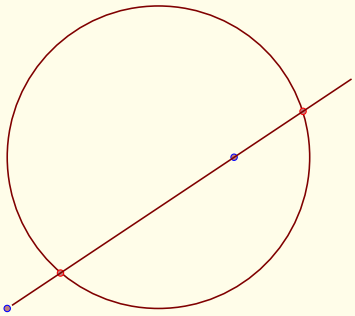


```
  \begin{tikzpicture}
  \tkzDefPoint(2,2){A}
  \tkzDefPoint(5,4){B}
  \tkzDefPoint(4,4){O}
  \FPeval{\tkzLen}{0.0002/0.0001}
  \tkzDrawCircle[R](O,\tkzLen cm)
  \tkzInterLC[R](A,B)(O, \tkzLen cm)
  \tkzGetPoints{I}{J}
  \tkzDrawPoints[color=blue](A,B)
  \tkzDrawPoints[color=red](I,J)
  \tkzDrawLine(I,J)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

10.2.7 Calcul de la mesure du rayon

Avec TEX et **\tkzLength**.

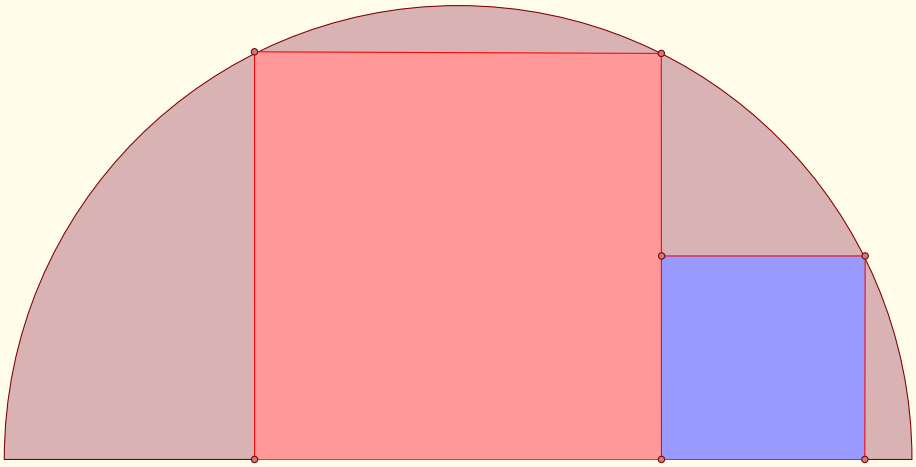
Cette dimension a été créée avec **\newdimen**. 2 cm a été transformé en points. Il est bien sûr possible d'utiliser TEX pour calculer.



```
\begin{tikzpicture}
  \tkzDefPoint(2,2){A}
  \tkzDefPoint(5,4){B}
  \tkzDefPoint(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}
```

10.2.8 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=1.5]
 \tkzInit[xmax=8,ymax=5]\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=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}
```

10.3 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

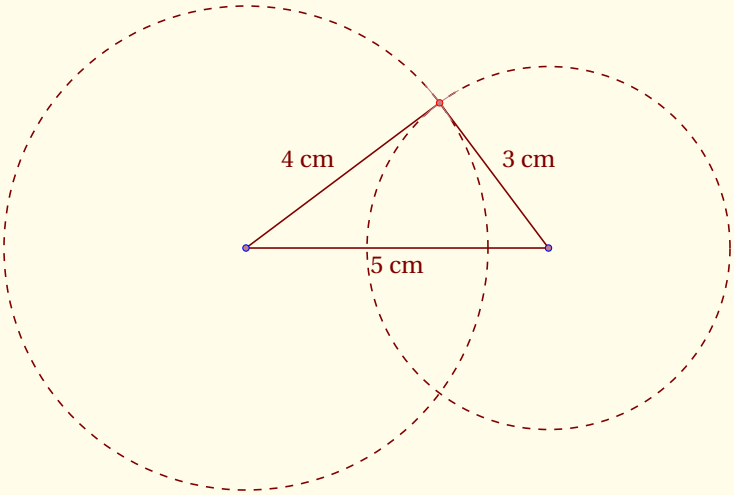
\tkzInterCC[⟨*options*⟩](⟨O,A/*r*⟩)(⟨O′,A′/*r*′⟩){⟨I⟩}{⟨J⟩}

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

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**.

10.3.1 Construction d'un triangle connaissant les mesures des côtés

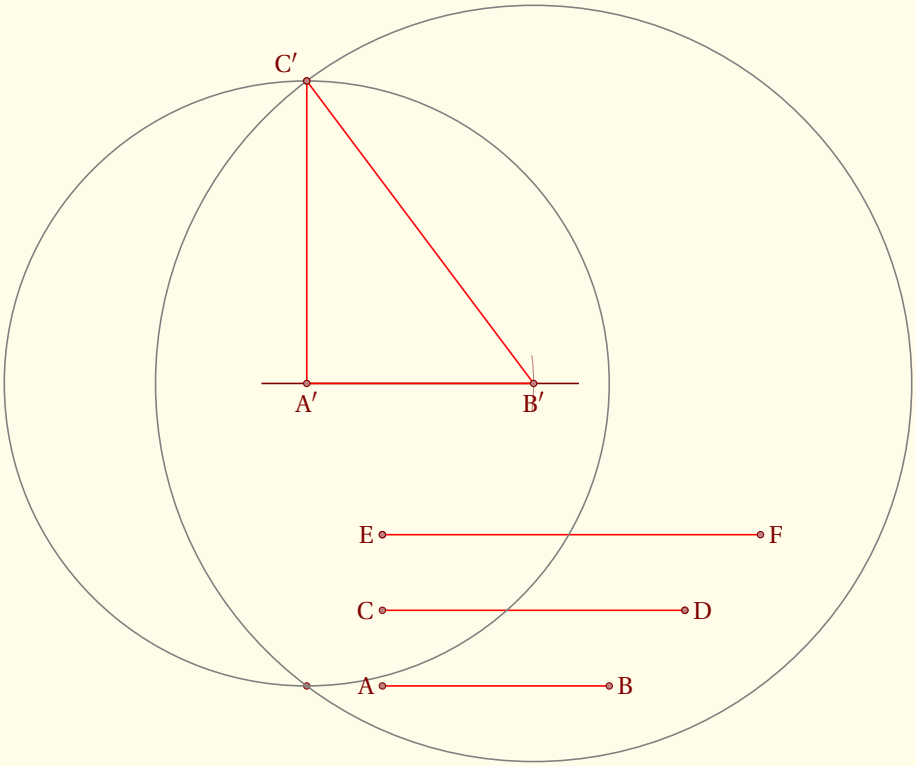
On veut obtenir le triangle de Pythagore (3,4,5)




```
\begin{tikzpicture}[scale=.8]
  \tkzDefPoint(0,0){A} \tkzDefPoint(5,0){B}
  \tkzDrawCircle[R,dashed](A,4 cm) \tkzDrawCircle[R,dashed](B,3 cm)
  \tkzInterCC[R](A,4 cm)(B,3 cm) \tkzGetPoints{C}{D}
  \tkzDrawPolygon(A,B,C)
  \tkzCompasss(A,C B,C)
  \tkzLabelSegment[below](A,B){$5$ cm}
  \tkzLabelSegment[above left](A,C){$4$ cm}
  \tkzLabelSegment[above right](B,C){$3$ cm}
  \tkzDrawPoints[color=red](C)
  \tkzDrawPoints[color=blue](A,B)
\end{tikzpicture}
```

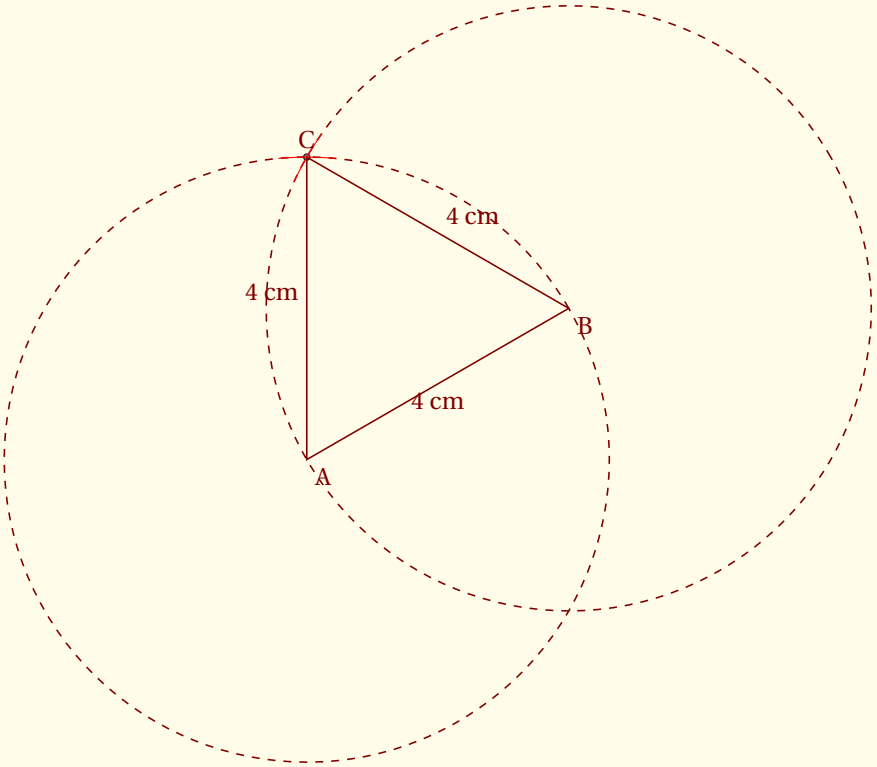
10.3.2 Dupliquer un triangle

Trois segments étant donnés, construire un triangle. Il s'agit de récupérer les mesures des longueurs avec **\tkzCalcLength**.



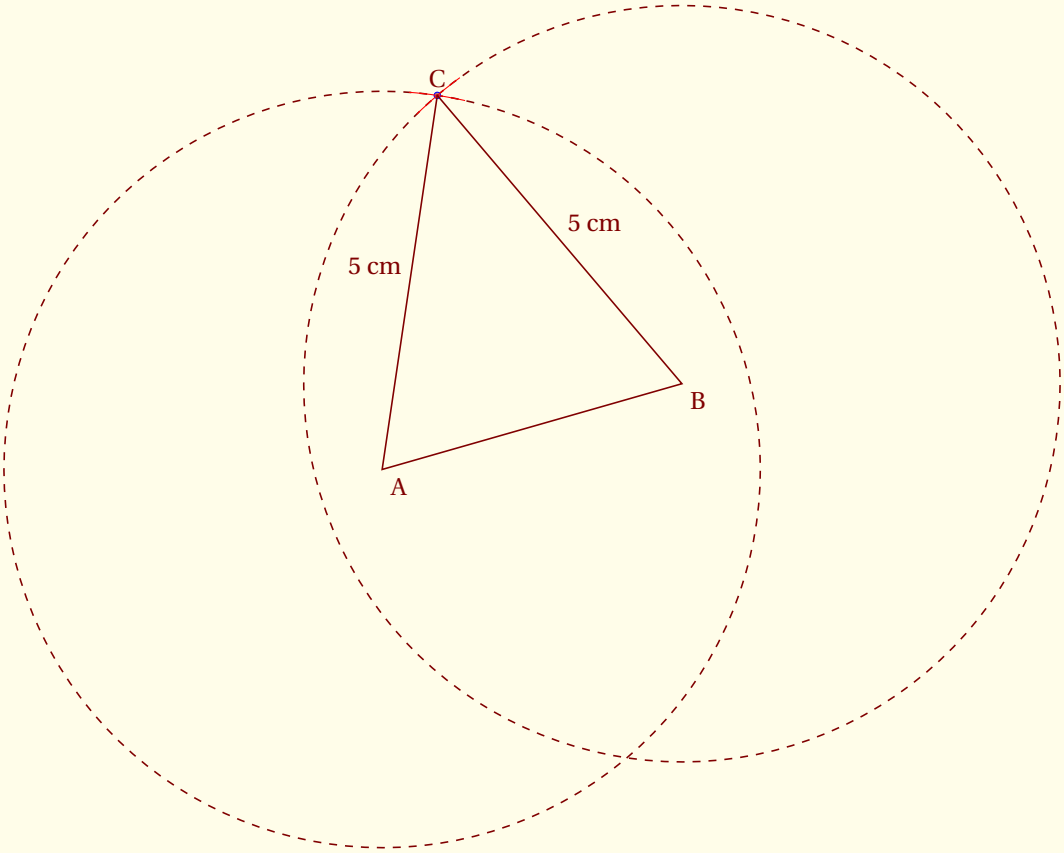

```
\begin{tikzpicture}
 \tkzDefPoint(1,0){A}  \tkzDefPoint(4,0){B}   % On place les points
 \tkzDefPoint(1,1){C}  \tkzDefPoint(5,1){D}
 \tkzDefPoint(1,2){E}  \tkzDefPoint(6,2){F}
 \tkzDefPoint(0,4){A'} \tkzDefPoint(3,4){B'}
 \tkzCalcLength[cm](C,D)\tkzGetLength{rCD}
 \tkzCalcLength[cm](E,F)\tkzGetLength{rEF}
 \tkzInterCC[R](A',\rCD cm)(B',\rEF cm)\tkzGetPoints{I}{J}
 \tkzDrawSegments[red](A,B C,D E,F) % Les tracés
 \tkzDrawLine(A',B')
 \tkzDrawPoints(D,E,I,J)
 \tkzDrawPolygon[color=red](A',B',I)
 \tkzSetUpLine[color=gray]
 \tkzCompass(A',B')
 \tkzDrawCircle[R](A',\rCD cm)
 \tkzDrawCircle[R](B',\rEF cm)
 \tkzDrawPoints(A,B,C,D,E,F,A',B',I)
 \tkzLabelPoints[left](A,C,E)
 \tkzLabelPoints[right](B,D,F)
 \tkzLabelPoints[below](A',B')
 \tkzLabelPoint[above left](I){$C'$}
\end{tikzpicture}
```

10.3.3 Construction d'un triangle équilatéral



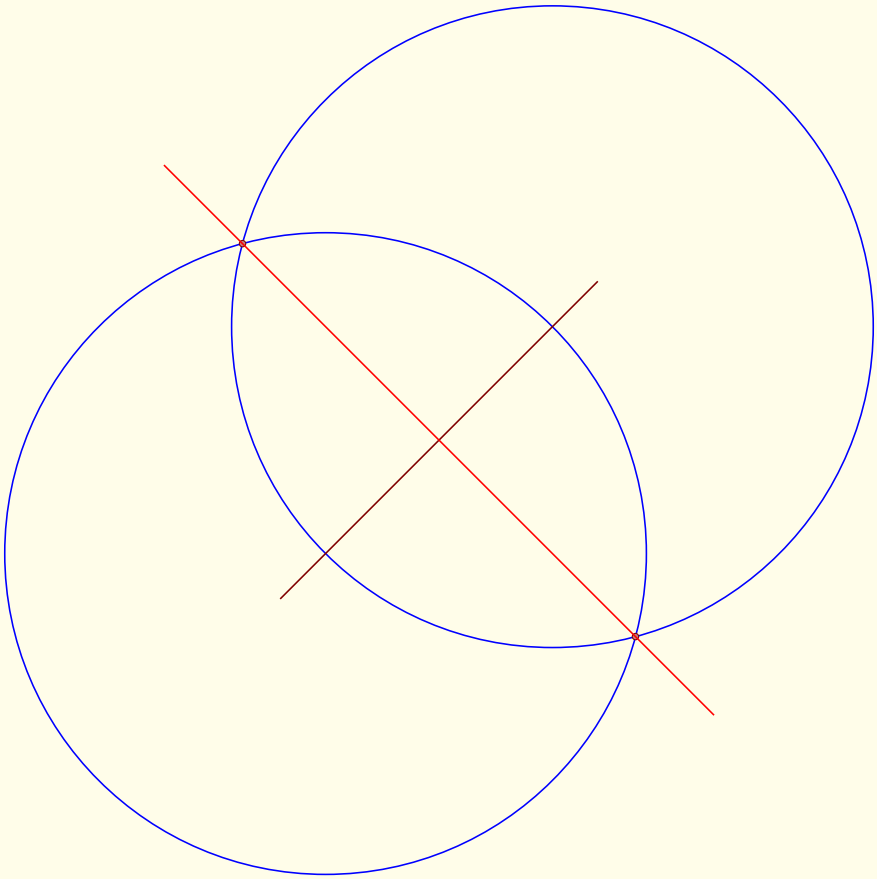

```
\begin{tikzpicture}[rotate=30]
 \tkzDefPoint(1,1){A}
 \tkzDefPoint(5,1){B}
 \tkzInterCC(A,B)(B,A)\tkzGetPoints{C}{D}
 \tkzDrawPoint[color=black](C)
 \tkzDrawCircle[dashed](A,B)
 \tkzDrawCircle[dashed](B,A)
 \tkzCompass[color=red](A,C)
 \tkzCompass[color=red](B,C)
 \tkzDrawPolygon(A,B,C)
 \tkzLabelSegment[above left](A,C){$4$ cm}
 \tkzLabelSegment[above right](B,C){$4$ cm}
 \tkzLabelSegment[below](A,B){$4$ cm}
 \tkzLabelPoints[](A,B)
 \tkzLabelPoint[above](C){$C$}
\end{tikzpicture}
```

10.3.4 Un triangle isocèle.




```
\begin{tikzpicture}[rotate=30]
 \tkzDefPoint(1,2){A}
 \tkzDefPoint(5,1){B}
 \tkzInterCC[R](A,5cm)(B,5cm)\tkzGetPoints{C}{D}
 \tkzDrawCircle[R,dashed](A,5 cm)
 \tkzDrawCircle[R,dashed](B,5 cm)
 \tkzDrawPoint[color=blue](C)
 \tkzCompass[color=red](A,C)
 \tkzCompass[color=red](B,C)
 \tkzDrawPolygon(A,B,C)
 \tkzLabelSegment[above left](A,C){$5$ cm}
 \tkzLabelSegment[above right](B,C){$5$ cm}
 \tkzLabelPoints[](A,B)
 \tkzLabelPoint[above](C){$C$}
\end{tikzpicture}
```

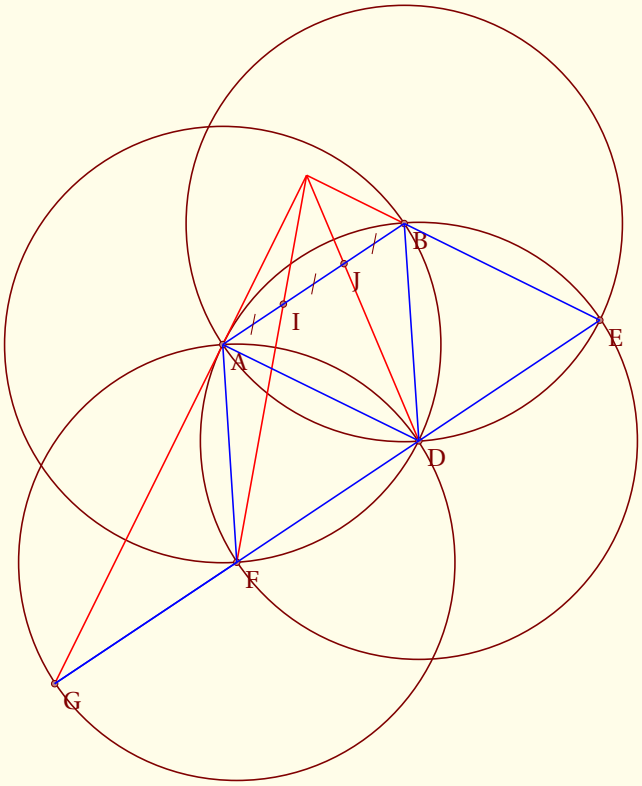
10.3.5 Exemple une médiatrice



```
\begin{tikzpicture}
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(3,3){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}
```

10.3.6 Trisection d'un segment

Voici un exemple complet utilisant toutes les macros précédentes. Il s'agit 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}
```

SECTION 11

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.

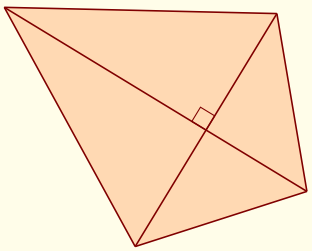
11.1 Définition de droites

\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**.

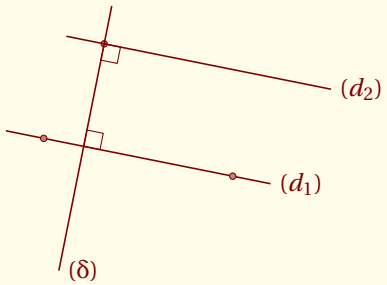
options	défaut	définition
mediator		médiatrice. Deux points sont définis
perpendicular=through...		perpendiculaire à une droite passant par un point
orthogonal=through...		voir ci-dessus
parallel=through...		parallèle à une droite passant par un point
bisector		bissectrice d'un angle défini par trois points
bisector out		bissectrice extérieure
K	1	Coefficient pour la droite perpendiculaire

11.1.1 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}
```

11.1.2 Exemple avec orthogonal et parallel




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

11.2 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}}}
```

Cela permet de tracer une partie d'une droite définie par deux points. On utilise pour cela deux valeurs, qui sont des pourcentages par rapport à la longueur du segment défini par les deux points.


```
\begin{tikzpicture}
   \tkzDefPoints{0/0/A,5/0/B}
   \tkzDrawLine[color=blue,thin, add=1 and 1,end   = $(\delta)$](A,B)
   \tkzDrawLine[color=red,thick, add=.5 and .5](A,B)
   \tkzDrawPoints(A,B)  \tkzLabelPoints(A,B)
    \tkzDrawLine[color=Maroon,line width=2pt, add=-.2 and -.2 ](A,B)
  \end{tikzpicture}
```

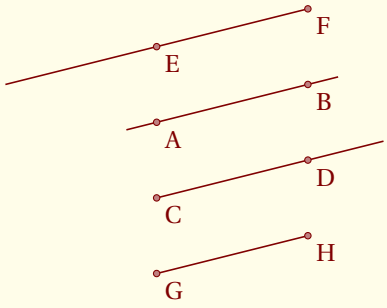
\tkzDrawLine[⟨*local options*⟩](⟨*pt1,pt2*⟩)

Les arguments sont une liste de deux points.

options	défaut	définition
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*

11.2.1 Exemple de tracer de droite avec add



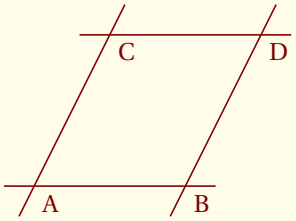

```
\begin{tikzpicture}
 \tkzInit[xmin=-2,xmax=3,ymin=-2.25,ymax=2.25]
 \tkzClip[space=.25]
 \tkzDefPoint(0,0){A} \tkzDefPoint(2,0.5){B}
 \tkzDefPoint(0,-1){C}\tkzDefPoint(2,-0.5){D}
 \tkzDefPoint(0,1){E} \tkzDefPoint(2,1.5){F}
 \tkzDefPoint(0,-2){G} \tkzDefPoint(2,-1.5){H}
  \tkzDrawLine(A,B)    \tkzDrawLine[add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

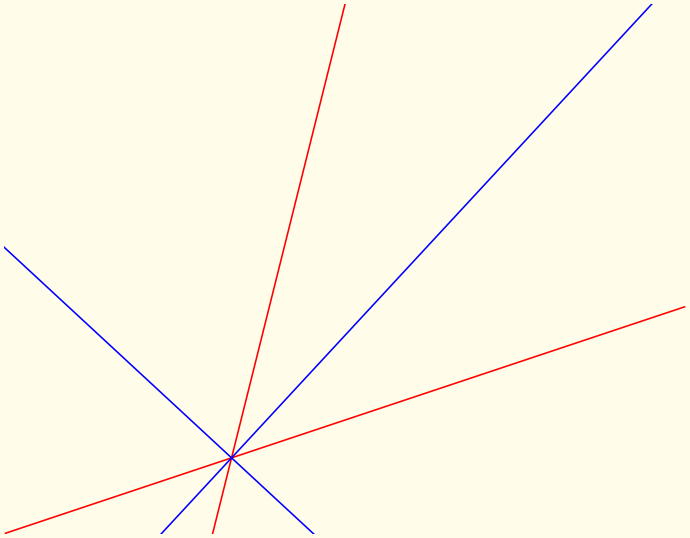
\tkzDrawLines[⟨*local options*⟩](⟨*pt1,pt2 pt3,pt4 ...*⟩)

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

11.2.2 Exemple avec \tkzDrawLines



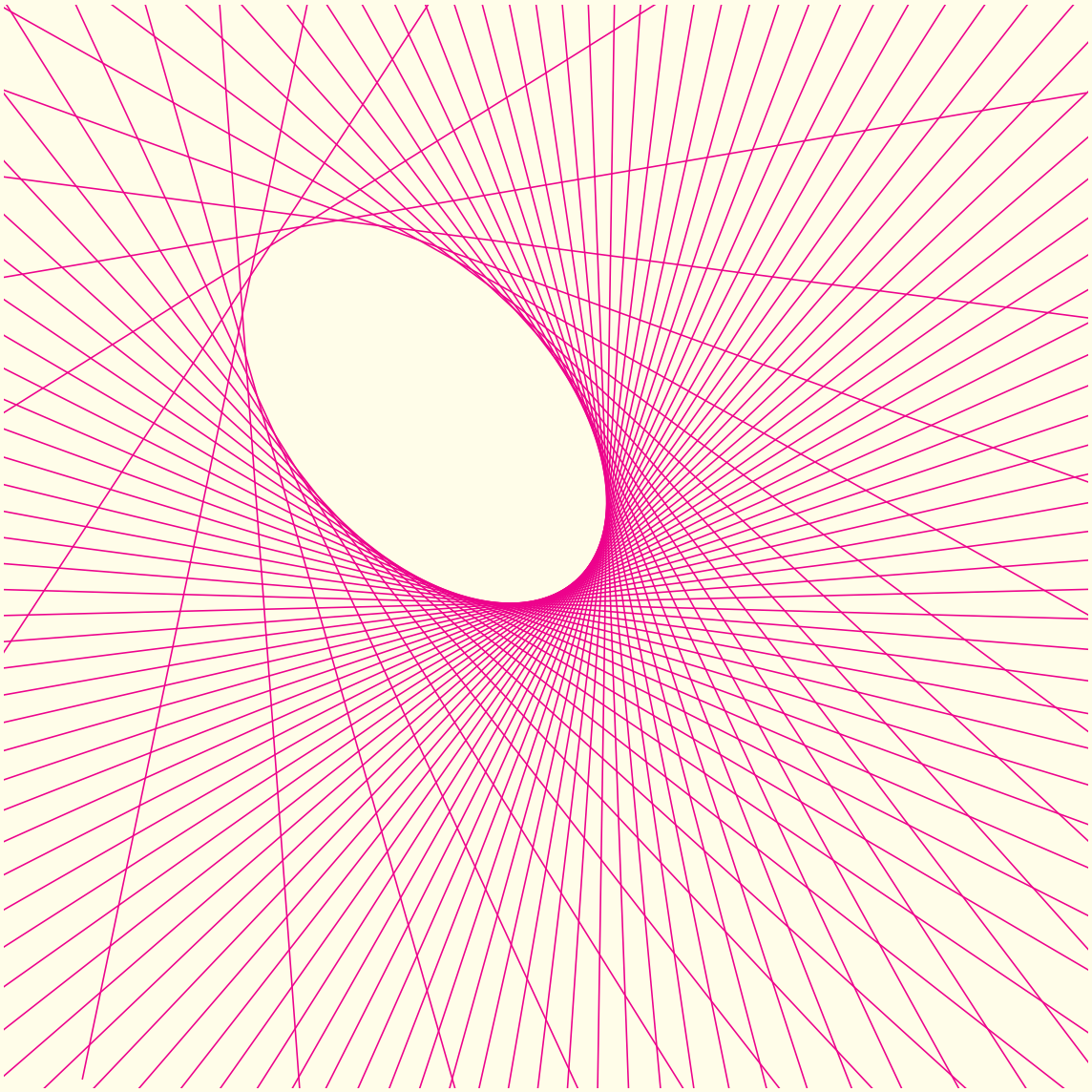

```
\begin{tikzpicture}
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(2,0){B}
  \tkzDefPoint(1,2){C}
  \tkzDefPoint(3,2){D}
  \tkzDrawLines(A,B C,D A,C B,D)
  \tkzLabelPoints(A,B,C,D)
\end{tikzpicture}
```



```
\begin{tikzpicture}
 \tkzInit[xmin=-3,xmax=6, ymin=-1,ymax=6]
 \tkzClip
 \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 1,color=red](O,I O,J)
 \tkzDrawLines[add = 5 and 5,color=blue](O,i O,j)
\end{tikzpicture}
```

11.2.3 Une enveloppe

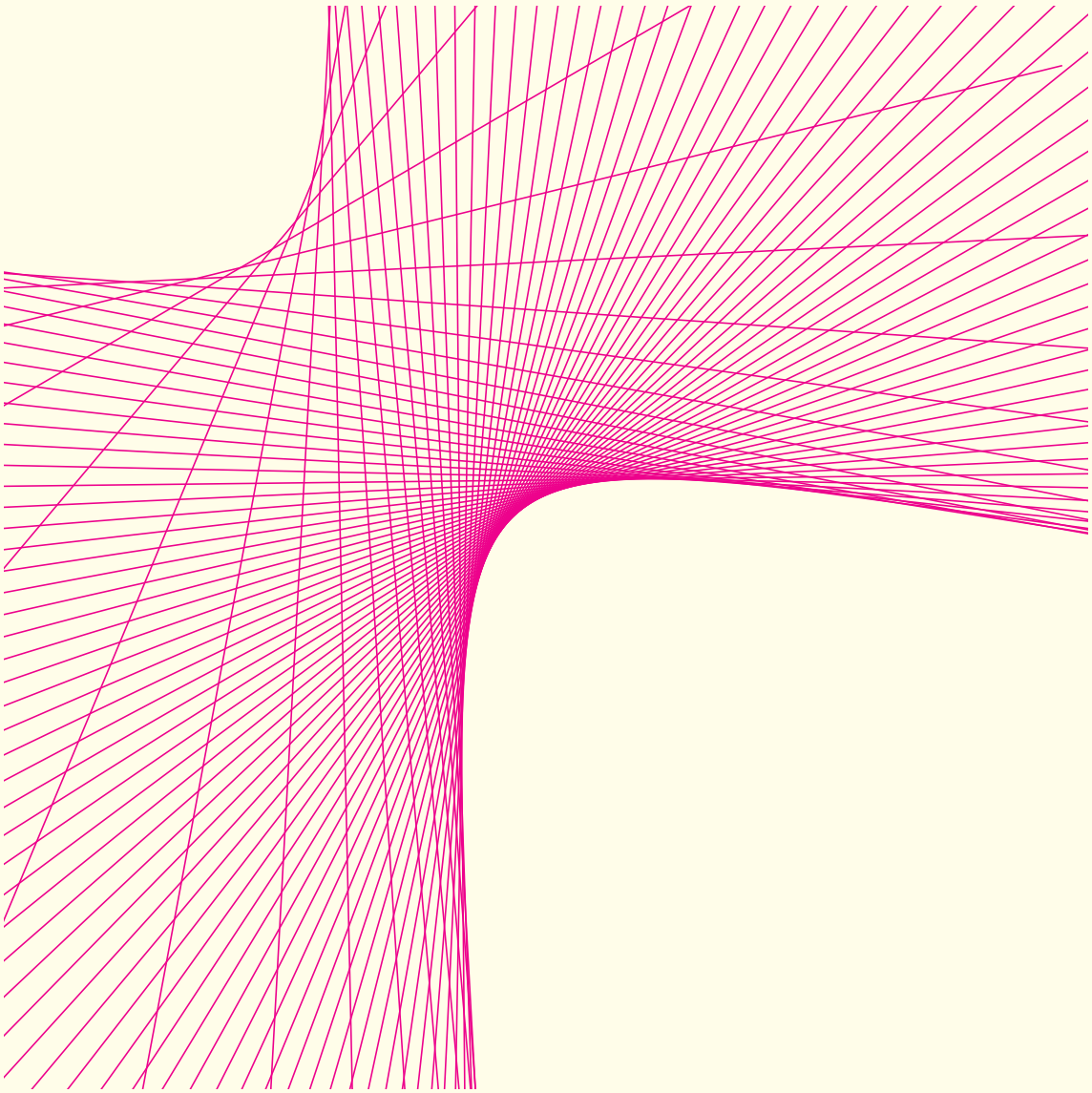
D'après une figure d'O. Reboux avec pst-eucl de D Rodriguez



```
\begin{tikzpicture}[scale=1.25]
  \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}
```

11.2.4 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}
```

11.3 Ajouter des labels aux droites \tkzLabelLine

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

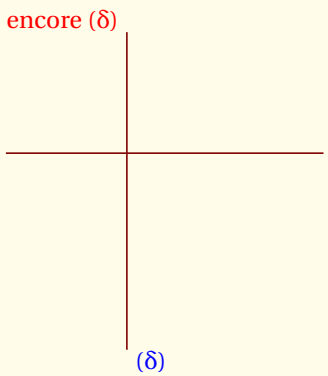
arguments	défaut	définition
label		exemple **\tkzLabelLine(A,B){δ}**

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****, . . .*

11.3.1 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}
   \tkzInit[ymin=-1,ymax=1.5,xmin=-2,xmax=2.5]
   \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}
```

11.4 Configurer les options pour les lignes \tkzSetUpLine

voir 20.2

11.5 Montrer les constructions de certaines lignes \tkzShowLine

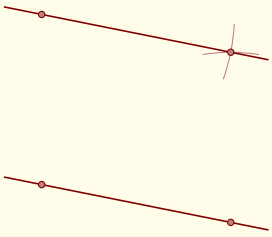
\tkzShowLine[⟨*local options*⟩](⟨*pt1,pt2*⟩) ou (⟨*pt1,pt2,pt3*⟩)

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)

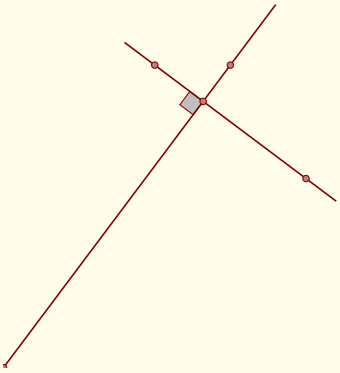
Il faut ajouter bien sûr tous les styles de ***TikZ*** *pour les tracés*

11.5.1 Exemple de \tkzShowLine et parallel



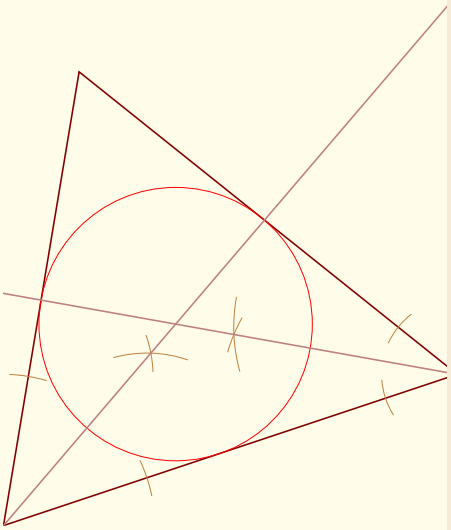
```
\begin{tikzpicture}
    \tkzDefPoints{-1.5/-0.25/A,1/-0.75/B,-1.5/2/C}
    \tkzDrawLine(A,B)
    \tkzDefLine[parallel=through C](A,B)  \tkzGetPoint{c}
    \tkzShowLine[parallel=through C](A,B)
    \tkzDrawLine(C,c)
    \tkzDrawPoints(A,B,C,c)
\end{tikzpicture}
```

11.5.2 Exemple de \tkzShowLine et perpendicular



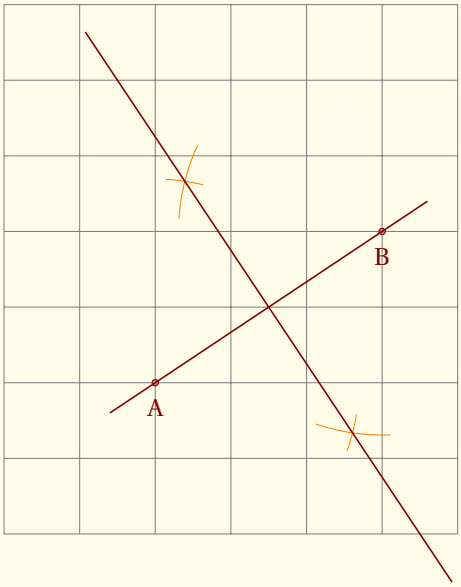
```
\begin{tikzpicture}
  \tkzInit[xmin=0,xmax=6,ymin=0,ymax=6]
  \tkzClip
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(3,4){B}
  \tkzDefPoint(2,4){C}
  \tkzDefLine[perpendicular=through C,%
              K=-.5](A,B)
  \tkzGetPoint{c}
  \tkzDefPointBy[projection=onto A--B](c)
  \tkzGetPoint{h}
  \tkzMarkRightAngle[fill=lightgray](A,h,C)
  \tkzDrawLines[](A,B C,c)
  \tkzDrawPoints(A,B,C,h,c)
\end{tikzpicture}
```

11.5.3 Exemple de \tkzShowLine et bisector



```
\begin{tikzpicture}
 \tkzInit[xmin=0,xmax=7,ymin=0,ymax=7]
 \tkzClip
 \tkzDefPoints{0/0/A, 6/2/B, 1/6/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}
 \tkzShowLine[bisector,size=2,gap=3](B,A,C)
 \tkzShowLine[bisector,size=1,gap=3](C,B,A)
 \tkzInterLL(A,a)(B,b) \tkzGetPoint{I}
 \tkzDefPointBy[projection = onto A--B](I)
 \tkzDrawCircle[radius,color=red,%
 line width=.2pt](I,tkzPointResult)
 \tkzDrawSegments[color=Maroon!50](I,tkzPointResult)
 \tkzDrawLines[add=0 and 5,color=Maroon!50](A,a B,b)
\end{tikzpicture}
```

11.5.4 Exemple de \tkzShowLine et mediator




```
\begin{tikzpicture}
 \tkzInit[xmax=6,ymax=7]
 \tkzGrid
 \tkzDefPoint(2,2){A}
 \tkzDefPoint(5,4){B}
 \tkzDrawPoints(A,B)
 \tkzShowLine[mediator,color=orange,length=1](A,B)
 \tkzGetPoints{i}{j}
 \tkzLabelPoints[below =3pt](A,B)
 \tkzDrawLines[](A,B i,j)
\end{tikzpicture}
```

SECTION 12

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**) .

12.1 Tracer un segment \tkzDrawSegment

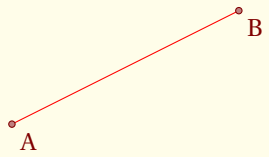
\tkzDrawSegment[⟨*local options*⟩](⟨*pt1,pt2*⟩)

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]

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

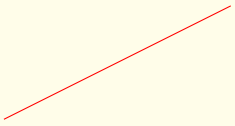
12.1.1 Exemple avec des références de points




```
\begin{tikzpicture}[scale=1.5]
  \tkzInit[xmin=-1,xmax=3,ymin=-1,ymax=2]
  \tkzClip
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(2,1){B}
  \tkzDrawSegment[color=red,thin](A,B)
  \tkzDrawPoints(A,B)
  \tkzLabelPoints(A,B)
\end{tikzpicture}
```

12.1.2 Exemple avec des références de points

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



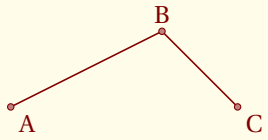
```
\begin{tikzpicture}[scale=1.5]
  \tkzInit[xmin=-1,xmax=3,ymin=-1,ymax=2]
  \tkzClip
  \tkzDrawSegment[color=red,thin]({0,0},{2,1})
\end{tikzpicture}
```

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

12.2 Tracer des segments \tkzDrawSegments

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

Les arguments sont une liste de couple de deux points. Les styles de **TikZ** *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}
```

12.3 Marquer un segment \tkzMarkSegment

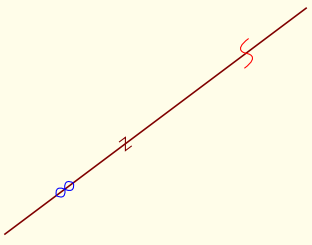
\tkzMarkSegment[⟨*local options*⟩](⟨*pt1,pt2*⟩)

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

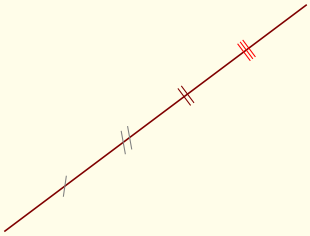
Les marques possibles sont celles fournies par **TikZ**, *mais d'autres marques ont été crées d'après une idée de Yves Combe.*

12.3.1 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}
```

12.3.2 Utilisation de mark



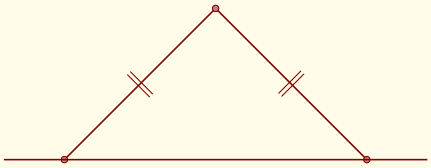
```
\begin{tikzpicture}
  \tkzDefPoint(2,1){A}
  \tkzDefPoint(6,4){B}
  \tkzDrawSegment(A,B)
  \tkzMarkSegment[color=gray,
                  pos=0.2,mark=s|](A,B)
  \tkzMarkSegment[color=gray,
                  pos=0.4,mark=s||](A,B)
  \tkzMarkSegment[color=Maroon,
                  pos=0.6,mark=|||(A,B)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

12.4 Marquer des segments \tkzMarkSegments

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

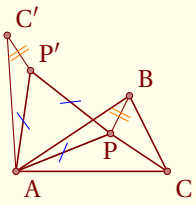
Les arguments sont une liste de couple de deux points séparés par des espaces. Les styles de ***TikZ*** *sont accessibles pour les tracés.*

12.4.1 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}
```

12.5 Exemple de rotation




```
\begin{tikzpicture}[scale=0.5]
 \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|,mark 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

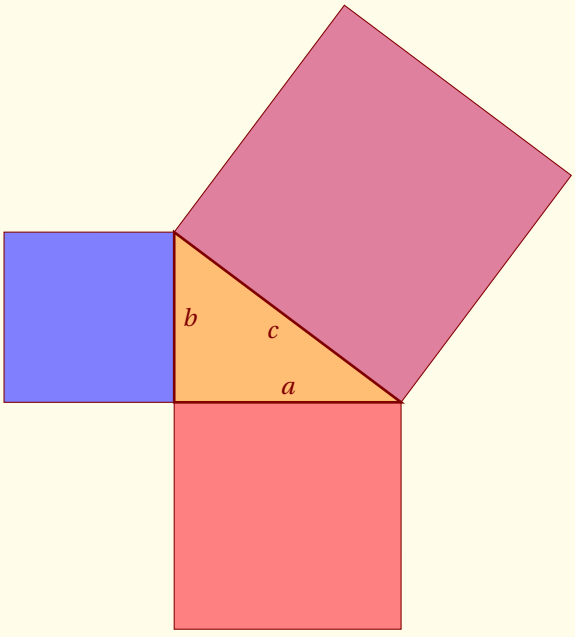
12.5.1 Labels multiples

$$\frac{a}{4}$$

```
\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}
```

12.5.2 Labels et Pythagore

Cet exemple nécessite **\usetkzobjpolygons**

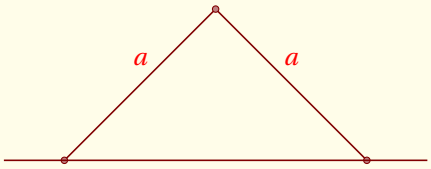



```
\begin{tikzpicture}[scale=.75]
\tkzInit[xmax=5,ymax=5]
\tkzDefPoint(0,0){C}
\tkzDefPoint(4,0){A}
\tkzDefPoint(0,3){B}
\tkzDefSquare(B,A)\tkzGetPoints{E}{F}
\tkzDefSquare(A,C)\tkzGetPoints{G}{H}
\tkzDefSquare(C,B)\tkzGetPoints{I}{J}
\tkzFillPolygon[draw,
                fill = red!50 ](A,C,G,H)
\tkzFillPolygon[draw,
                fill = blue!50 ](C,B,I,J)
\tkzFillPolygon[draw,
                fill = purple!50](B,A,E,F)
\tkzFillPolygon[draw,opacity=.5,
                fill = orange](A,B,C)
\tkzDrawPolygon[line width = 1pt](A,B,C)
\tkzLabelSegment[above](C,A){$a$}
\tkzLabelSegment[right](B,C){$b$}
\tkzLabelSegment[below left](B,A){$c$}
\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.*

12.5.3 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}
```

SECTION 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.

\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**

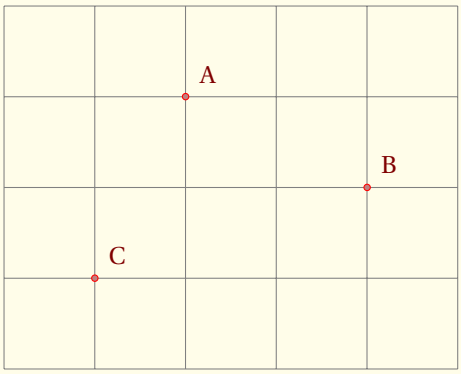
Dans ce qui suit, on suppose que le point est récupéré par **\tkzGetPoint{C}**

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)	$\overrightarrow{CD} = \overrightarrow{AB}$
K	[linear](A,B),K=2	$\overrightarrow{AC} = 2 \times \overrightarrow{AB}$

Pour la linéarité, K est obligatoire. Sa valeur par défaut est égale à 1.

13.1.1 \tkzDefPointWith et orthogonal

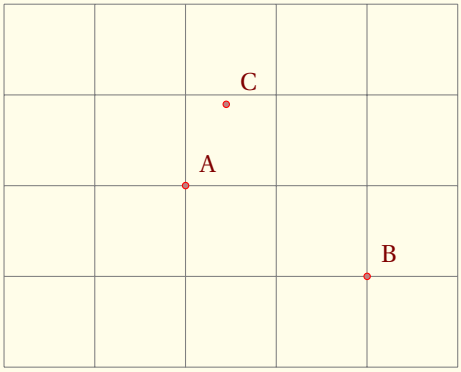
$K = -1$ c'est pour que $(\overrightarrow{AC}, \overrightarrow{AB})$ détermine un angle positif. AB=AC puisque $K = 1$




```
\begin{tikzpicture}[scale=1.2]
   \tkzInit[xmax=5,ymax=4] \tkzGrid
   \tkzDefPoint(2,3){A}   \tkzDefPoint(4,2){B}
   \tkzDefPointWith[orthogonal,K=-1](A,B)
   \tkzGetPoint{C}
   \tkzDrawPoints[color=red](A,B,C)
   \tkzLabelPoints[above right=3pt](A,B,C)
\end{tikzpicture}
```

13.1.2 \tkzDefPointWith orthogonal normed

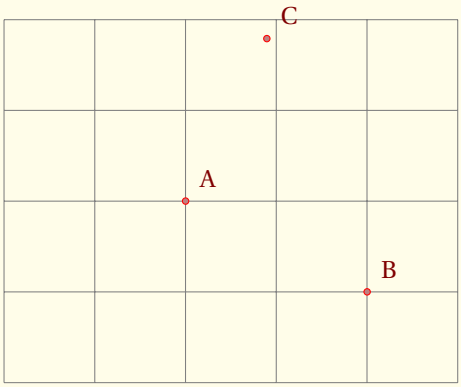
AC=1




```
\begin{tikzpicture}[scale=1.2]
   \tkzInit[ymin=1,xmax=5,ymax=5] \tkzGrid
   \tkzDefPoint(2,3){A}   \tkzDefPoint(4,2){B}
   \tkzDefPointWith[orthogonal normed](A,B)
   \tkzGetPoint{C}
   \tkzDrawPoints[color=red](A,B,C)
   \tkzLabelPoints[above right=3pt](A,B,C)
\end{tikzpicture}
```

13.1.3 \tkzDefPointWith et orthogonal normed

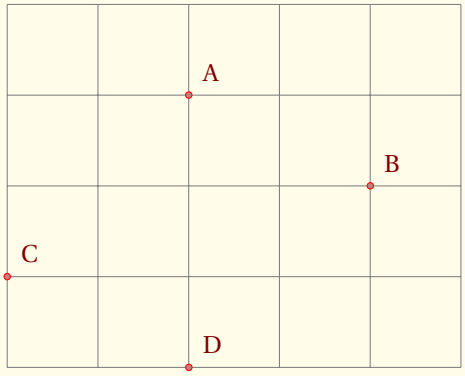
K = 2 donc AC=2.




```
\begin{tikzpicture}[scale=1.2]
   \tkzInit[ymin=1,xmax=5,ymax=5] \tkzGrid
   \tkzDefPoint(2,3){A}   \tkzDefPoint(4,2){B}
   \tkzDefPointWith[orthogonal normed,K=2](A,B)
   \tkzGetPoint{C}
   \tkzDrawPoints[color=red](A,B,C)
   \tkzLabelPoints[above right=3pt](A,B,C)
\end{tikzpicture}
```

13.1.4 \tkzDefPointWith et colinear

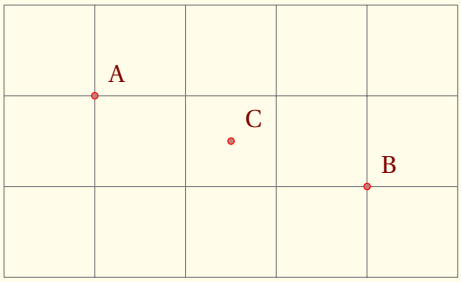
K = 2 donc AC=2.




```
\begin{tikzpicture}[scale=1.2]
   \tkzInit[xmax=5,ymax=4] \tkzGrid
   \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)
\end{tikzpicture}
```

13.1.5 \tkzDefPointWith linear

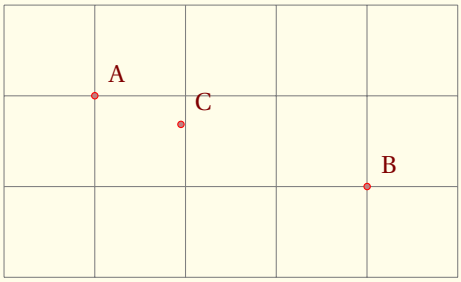
Ici K = 0.5 Cela revient à appliquer une homothétie ou bien encore une multiplication d'un vecteur par un réel. C est ici le milieu de [AB].




```
\begin{tikzpicture}[scale=1.2]
   \tkzInit[ymin=1,xmax=5,ymax=4] \tkzGrid
   \tkzDefPoint(1,3){A}   \tkzDefPoint(4,2){B}
   \tkzDefPointWith[linear,K=0.5](A,B)
   \tkzGetPoint{C}
   \tkzDrawPoints[color=red](A,B,C)
   \tkzLabelPoints[above right=3pt](A,B,C)
\end{tikzpicture}
```

13.1.6 \tkzDefPointWith linear normed

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




```
\begin{tikzpicture}[scale=1.2]
   \tkzInit[ymin=1,xmax=5,ymax=4] \tkzGrid
   \tkzDefPoint(1,3){A}   \tkzDefPoint(4,2){B}
   \tkzDefPointWith[linear normed](A,B)
   \tkzGetPoint{C}
   \tkzDrawPoints[color=red](A,B,C)
   \tkzLabelPoints[above right=3pt](A,B,C)
\end{tikzpicture}
```

SECTION 14

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**.

14.1 Caractéristiques d'un cercle : \tkzDefCircle

Pour le moment, il est possible de récupérer les caractéristiques des cercles suivants (le premier est là pour que l'ensemble soit homogène)

- **radius** cercle caractérisé par deux points définissant un rayon,
- **diameter** cercle caractérisé par deux points définissant un diamètre,
- **circum** cercle circonscrit à un triangle,
- **in** cercle inscrit dans à un triangle,
- **euler** cercle d'Euler d'un triangle,
- **apollonius** cercle d'Apollonius caractérisé par un segment et un ratio.

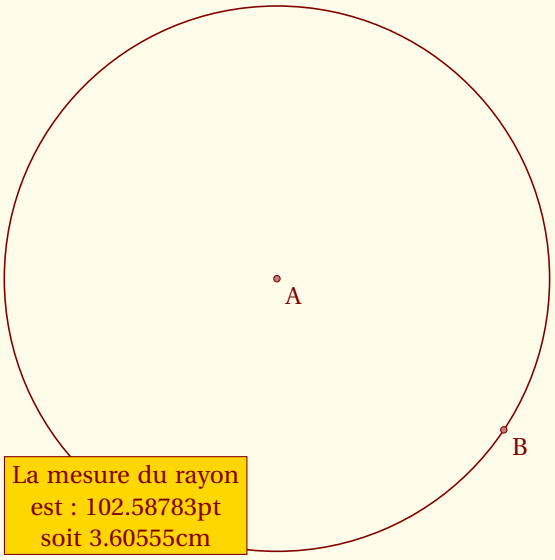
\tkzDefCircle[⟨*local options*⟩](⟨*A,B*⟩) ou (⟨*A,B,C*⟩)

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
radius	radius	cercle caractérisé par deux points définissant un rayon
diameter	radius	cercle caractérisé par deux points définissant un diamètre
circum	radius	cercle circonscrit à un triangle
in	radius	cercle inscrit dans à un triangle
euler	radius	Cercle d'Euler
apollonius	radius	Cercle d'Apollonius
orthogonal	radius	Cercle de centre donné orthogonal à un autre cercle
orthogonal through	radius	Cercle orthogonal à un autre cercle passant par deux points
K	2	Coefficient utilisé pour un cercle d'Apollonius
color	black	couleur du cercle
fill		couleur du disque, si présent
line width	.4pt	épaisseur du trait

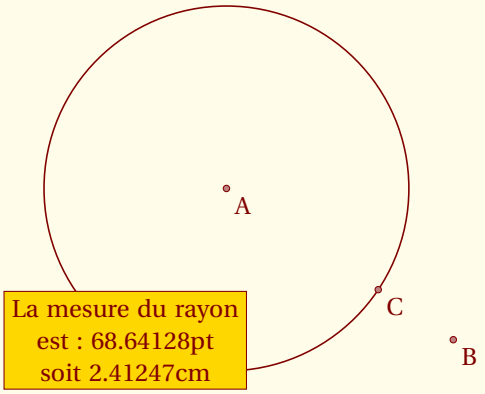
Dans les exemples suivants, je trace les cercles avec une macro pas encore présentée, mais ce n'est pas nécessaire. Dans certains cas on peut seulement avoir besoin du centre ou encore du rayon.

14.1.1 Exemple



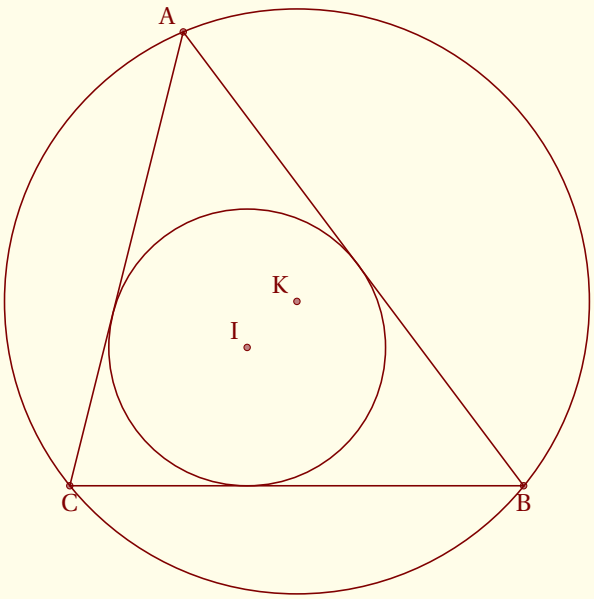

```
\begin{tikzpicture}
   \tkzDefPoint(0,4){A}
   \tkzDefPoint(3,2){B}
   \tkzDefCircle[radius](A,B)
   \tkzGetLength{rABpt}
   \tkzpttocm(\rABpt){rABcm}
   \tkzDrawCircle(A,B)
   \tkzDrawPoints(A,B)
   \tkzLabelPoints(A,B)
   \tkzLabelCircle[draw,fill=Gold,%
    text width=3cm,text centered](A,B)(-90)%
  {La mesure du rayon est :
  \rABpt pt soit \rABcm cm}
\end{tikzpicture}
```

14.1.2 Exemple avec un point aléatoire




```
\begin{tikzpicture}
   \tkzDefPoint(0,4){A}
   \tkzDefPoint(3,2){B}
   \tkzDefMidPoint(A,B) \tkzGetPoint{I}
   \tkzGetRandPointOn[segment = I--B]{C}
   \tkzDefCircle[radius](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](A,C)(-90)%
   {La mesure du rayon est :
    \rACpt pt soit \rACcm cm}
\end{tikzpicture}
```

14.1.3 Cercles inscrit et circonscrit pour un triangle donné

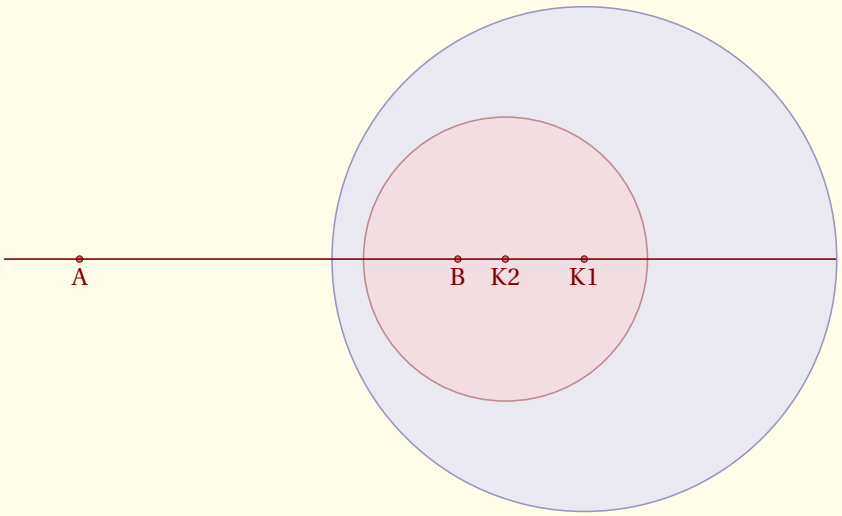



```
\begin{tikzpicture}[scale=1.5]
   \tkzDefPoint(2,2){A}
   \tkzDefPoint(5,-2){B}
   \tkzDefPoint(1,-2){C}
   \tkzDefCircle[in](A,B,C)
   \tkzGetPoint{I}    \tkzGetLength{rIN}
   \tkzDefCircle[circum](A,B,C)
   \tkzGetPoint{K}   \tkzGetLength{rCI}
   \tkzDrawPoints(A,B,C,I,K)
   \tkzDrawCircle[R,blue](I,\rIN pt)
   \tkzDrawCircle[R,red](K,\rCI pt)
   \tkzLabelPoints[below](B,C)
   \tkzLabelPoints[above left](A,I,K)
   \tkzDrawPolygon(A,B,C)
\end{tikzpicture}
```

14.1.4 Cercles d'Apollonius colorié pour un segment donné

Wikipedia donne comme définition :

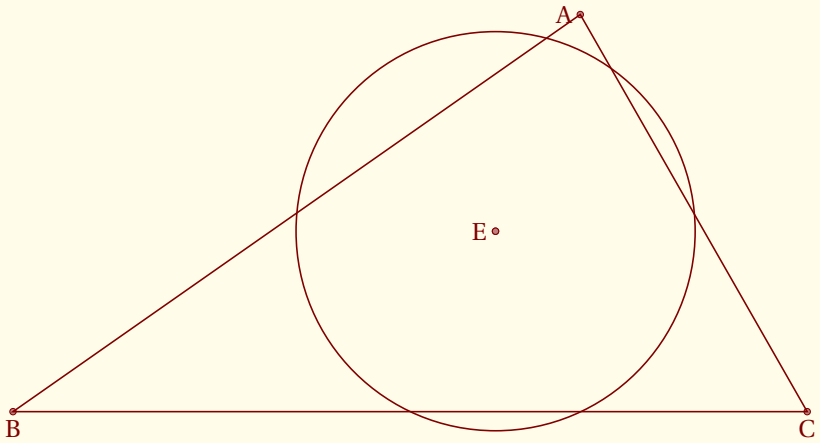
Apollonius de Perga propose de définir le cercle comme l'ensemble des points M du plan pour lesquels le rapport des distances MA/MB reste constant, les points A et B étant donnés. Théorème — Si A et B sont deux points distincts et k est un réel autre que 0 et 1, le cercle d'Apollonius du triplet (A,B,k) est l'ensemble des points M du plan tels que MA/MB = k.




```
\begin{tikzpicture}[scale=1.25]
  \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}
```

Les cercles ont été tracés et les disques coloriés, simplement avec les outils de **TikZ**.

14.1.5 Cercle d'Euler pour un triangle donné

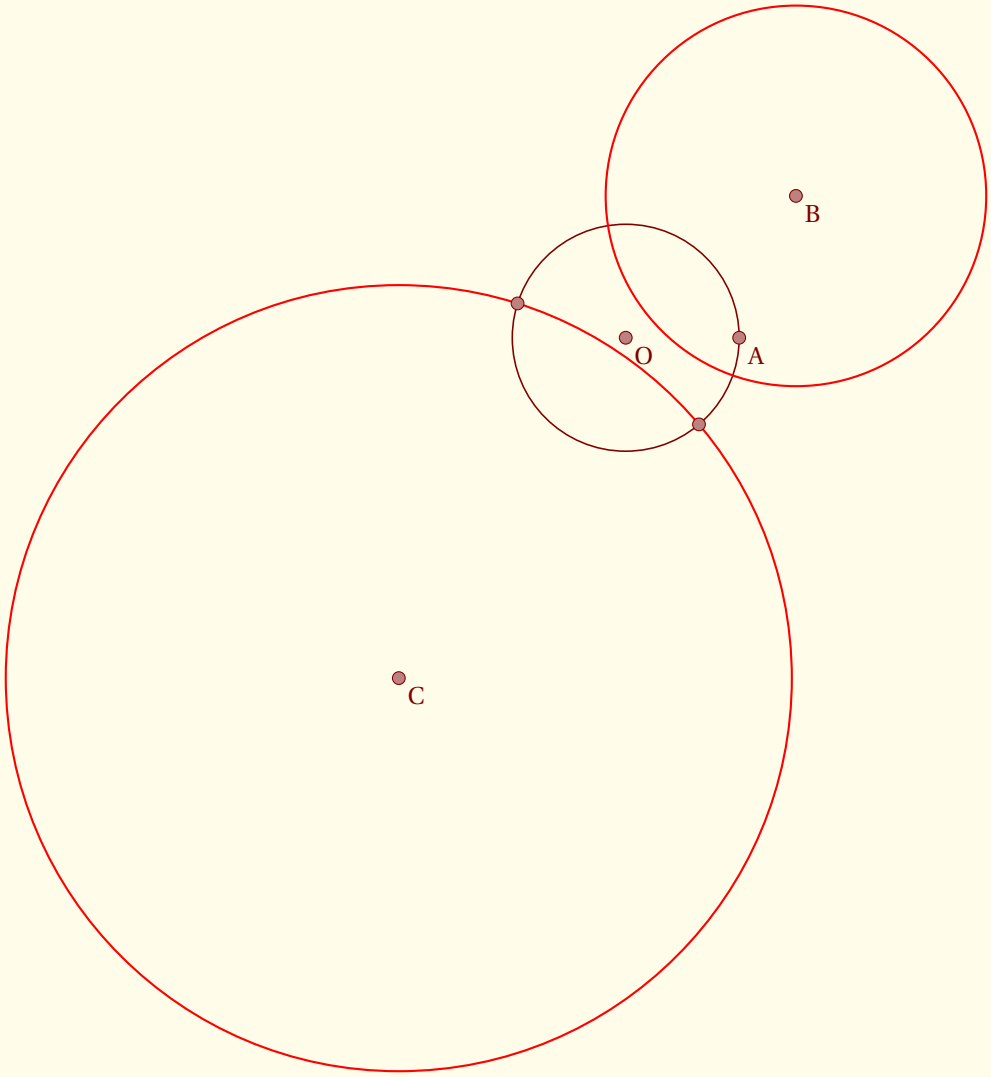



```
\begin{tikzpicture}[scale=1.5]
   \tkzInit[xmin=-1,ymin=-1,xmax=8,ymax=6] \tkzClip
   \tkzDefPoint(5,3.5){A} \tkzDefPoint(0,0){B} \tkzDefPoint(7,0){C}
   \tkzDefCircle[euler](A,B,C)
   \tkzGetPoint{E}  \tkzGetLength{rEuler}
   \tkzDrawPoints(A,B,C,E)
   \tkzDrawCircle[R,blue](E,\rEuler pt)
   \tkzDrawPolygon(A,B,C)
   \tkzLabelPoints[below](B,C)  \tkzLabelPoints[left](A,E)
\end{tikzpicture}
```

Il est possible avec les outils d'intersection de déterminer les points communs du cercle d'Euler et du triangle.

14.1.6 Cercle orthogonal de centre donné

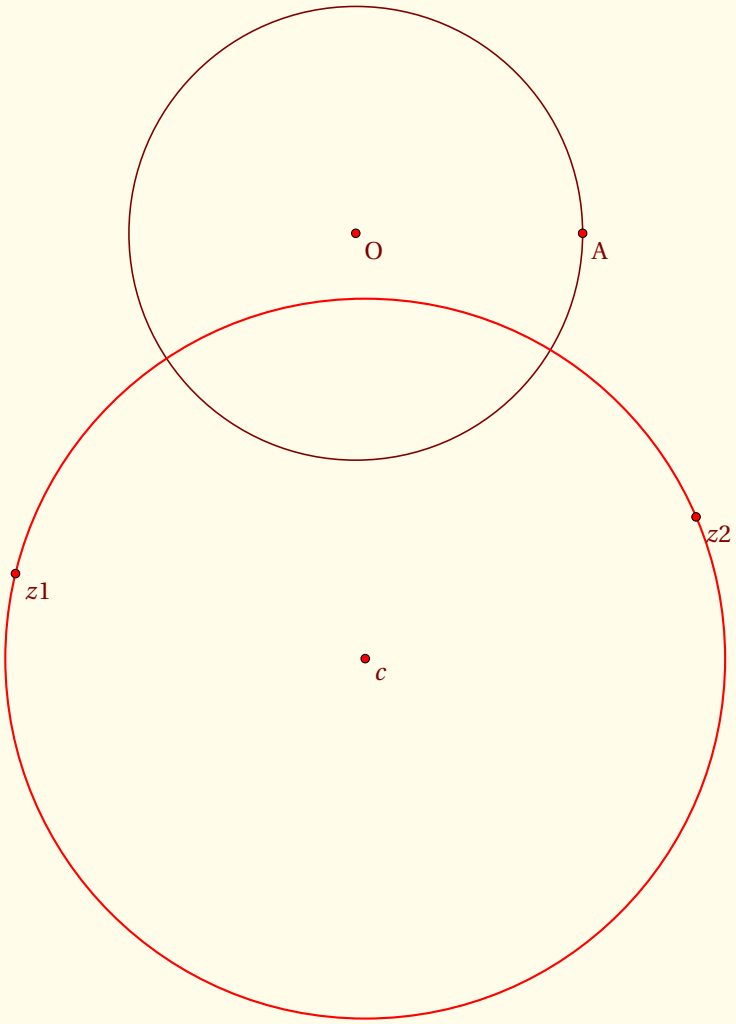
Nous allons chercher deux cercles orthogonaux au cercle de centre O passant par A, leurs centres B et C étant donnés.




```
\begin{tikzpicture}[scale=1.5]
  \tkzDefPoint(0,0){O}  \tkzDefPoint(1,0){A}
  \tkzDefPoint(1.5,1.25){B}  \tkzDefPoint(-2,-3){C}
  \tkzDrawCircle(O,A)
  \tkzDefCircle[orthogonal from=B](O,A)
  \tkzDrawCircle[thick,color=red](B,tkzFirstPointResult)
  \tkzDefCircle[orthogonal from=C](O,A)
  \tkzDrawCircle[thick,color=red](C,tkzFirstPointResult)
  \tkzDrawPoints(tkzFirstPointResult,tkzSecondPointResult,O,A,B,C)
  \tkzLabelPoints(O,A,C,B)
\end{tikzpicture}
```

14.1.7 Cercle orthogonal passant par deux points donnés

Nous allons cette fois récupéré le centre.




```
\begin{tikzpicture}[scale=3]
  \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}
```

14.2 Tracer un cercle

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

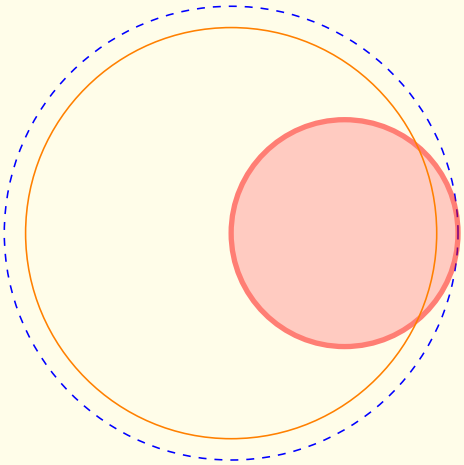
Attention les arguments sont des listes de deux ou bien de trois points. Les cercles que l'on peut tracer sont les mêmes que pour la macro précédente. Une option supplémentaire **R** *afin de donner directement une mesure.*

options	défaut	définition
radius	radius	cercle avec deux points définissant un rayon
diameter	radius	cercle avec deux points définissant un diamètre
R	radius	cercle caractérisé par un point et la mesure d'un rayon
circum	radius	cercle circonscrit à un triangle
in	radius	cercle inscrit dans à un triangle
euler	radius	Le cercle d'Euler
apollonius	radius	Le cercle d'Apollonius
K	2	Coefficient utilisé pour un cercle d'Apollonius
orthogonal	radius	Cercle de centre donné orthogonal à un autre cercle
orthogonal through	radius	Cercle orthogonal à un autre cercle passant par deux points

Il faut ajouter bien sûr tous les styles de **TikZ** *pour les tracés*

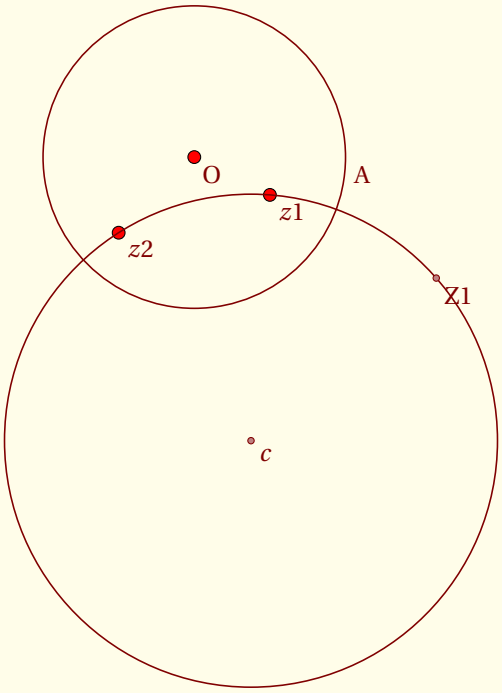
14.2.1 Cercles et styles, tracer un cercle et colorier le disque

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
  \FPeval\rayon{exp(1)}
  \tkzDrawCircle[R,color=orange](O,\rayon cm)
\end{tikzpicture}
```

14.2.2 Cercle orthogonal à un cercle donné passant par deux points donnés

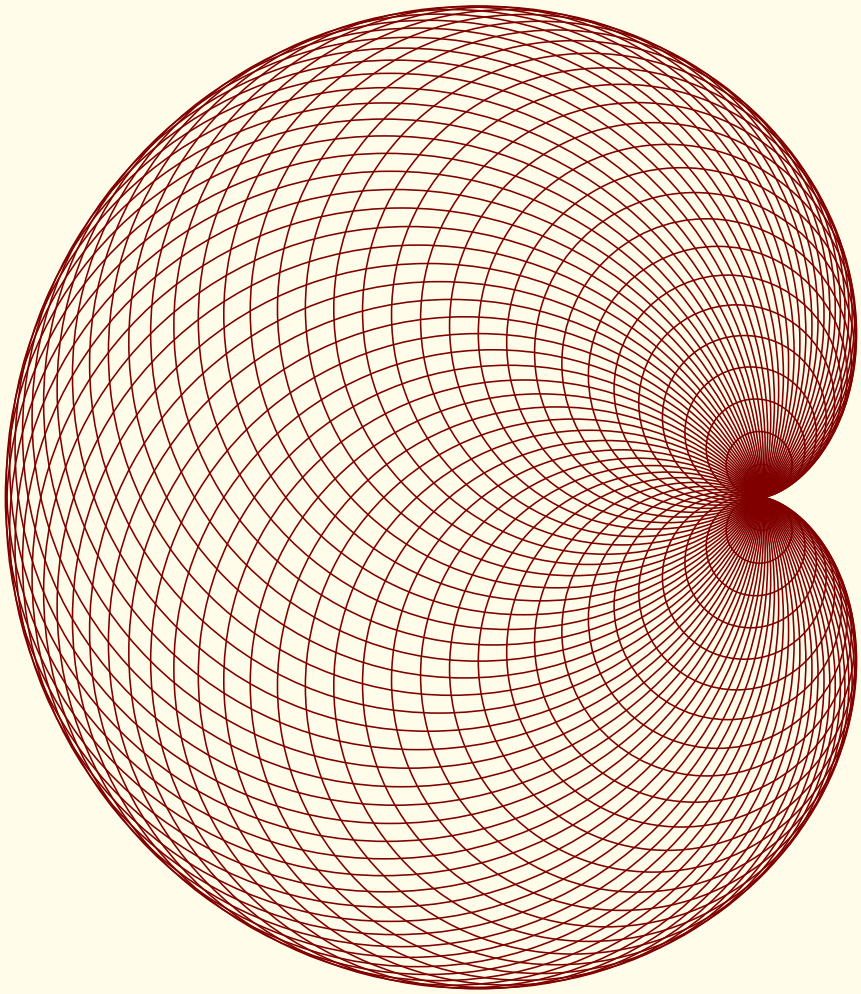



```
\begin{tikzpicture}[scale=2]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(1,0){A}
  \tkzDrawCircle(O,A)
  \tkzDefPoint(0.5,-0.25){z1}
  \tkzDefPoint(-0.5,-0.5){z2}
  \tkzDrawPoints[color = black,fill   = red,size=12](O,z1,z2)
  \tkzDefPointBy[inversion = center O through A](z1) \tkzGetPoint{Z1}
  \tkzCircumCenter(z1,z2,Z1) \tkzGetPoint{c}
  \tkzDrawCircle(c,Z1)
  \tkzDrawPoints(c,Z1)
  \tkzLabelPoints(O,A,z1,z2,Z1,c)
\end{tikzpicture}
```

14.2.3 Cardioïde

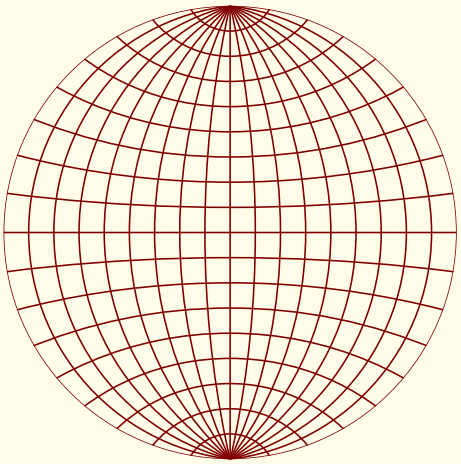
D'après une idée d'O. Reboux réalisée avec pst-eucl (module de Pstricks) de D. Rodriguez.

Son nom vient du grec kardia (cœur), en référence à sa forme, et lui fut donné par Johan Castillon. Wikipedia



```
\begin{tikzpicture}[scale=1.25]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2,0){A}
  \foreach \ang in {5,10,...,360}{%
     \tkzDefPoint(\ang:2){M}
     \tkzDrawCircle(M,A)
   }
\end{tikzpicture}
```

14.2.4 Ceci est une mappemonde



```
\begin{tikzpicture}[scale=.333]
 \tkzInit[xmin=-10,xmax=10,ymin=-10,ymax=10]
 \tkzDefPoint(0 , 0){O}
 \tkzDefPoint(9 , 0){A}
 \tkzDefPoint(-9, 0){C}
 \tkzDefPoint(0 , 9){B}
 \tkzDefPoint(0 ,-9){D}
 \tkzClipCircle(O,A)
 \foreach \pti in {1,2,...,8}{
 \tkzDefPoint(10*\pti:9){P\pti}
 \tkzDefPoint(90:\pti){MP\pti}
 \tkzDefPoint(0: \pti){NP\pti}
 \tkzDefLine[mediator](MP\pti,P\pti)
 \tkzInterLL(B,D)(tkzFirstPointResult,tkzSecondPointResult)
 \tkzDrawCircle[color=Maroon](tkzPointResult,P\pti)
 }
 \foreach \pti in {-1,-2,...,-8}{
 \tkzDefPoint(10*\pti:9){P\pti}
 \tkzDefPoint(-90:-\pti){MP\pti}
 \tkzDefPoint(0: -\pti){NP\pti}
 \tkzDefLine[mediator](MP\pti,P\pti)
 \tkzInterLL(B,D)(tkzFirstPointResult,tkzSecondPointResult)
 \tkzDrawCircle[color=Maroon](tkzPointResult,P\pti)
 }
 \foreach \pti in {1,2,...,8}{
 \tkzDefLine[mediator](B,NP\pti)
 \tkzInterLL(A,C)(tkzFirstPointResult,tkzSecondPointResult)
 \tkzDrawCircle[color=Maroon](tkzPointResult,NP\pti)
 }
 \foreach \pti in {1,2,...,8}{
 \tkzDefPoint(0: -\pti){NP\pti}
 \tkzDefLine[mediator](B,NP\pti)
 \tkzInterLL(A,C)(tkzFirstPointResult,tkzSecondPointResult)
 \tkzDrawCircle[color=Maroon](tkzPointResult,NP\pti)
 }
  \tkzDrawCircle[R,color=Maroon](O,9 cm)
  \tkzDrawSegments[color=Maroon](A,C B,D)
\end{tikzpicture}
```

14.3 Colorier un disque

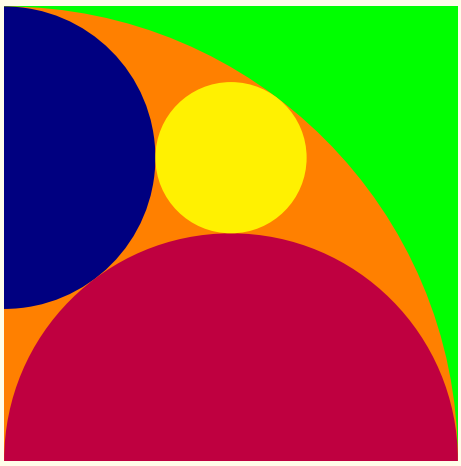
C'était possible avec la macro précédente, mais le tracé du disque était obligatoire, là ce n'est plus le cas.

\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* **TikZ***pour les tracés*

14.3.1 Exemple de \tkzFillCircle provenant d'un sangaku



```
\begin{tikzpicture}
 \tkzInit[xmin=0,xmax = 6,ymin=0,ymax=6] \tkzClip
 \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}
 \tkzTangent[from = B](F,A) \tkzGetPoints{G}{H}
 % \tkzTgtFromP(F,A)(B) est obsolète
 \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}
```

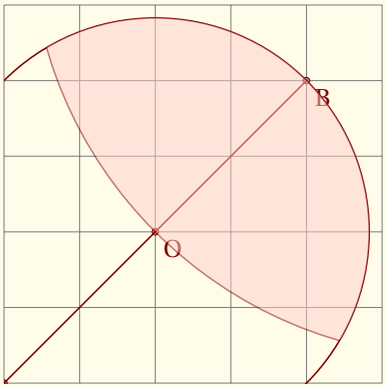
14.4 Clipper un disque

\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

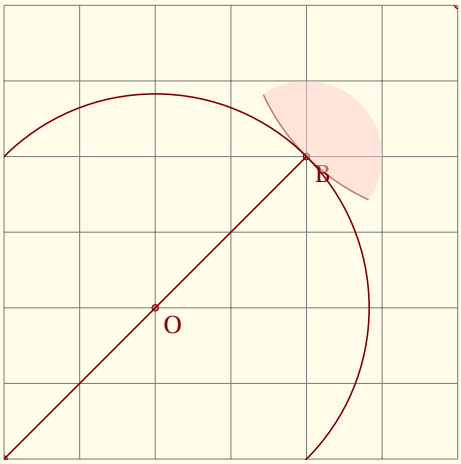
Il n'est pas nécessaire de mettre ***radius*** *car c'est l'option par défaut.*

14.4.1 Exemple 1 de \tkzClipCircle



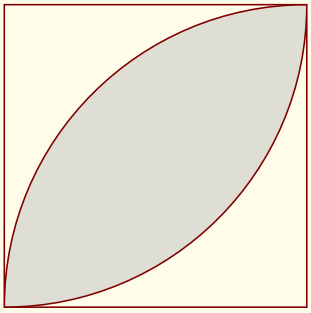

```
  \begin{tikzpicture}
  \tkzInit[xmax=5,ymax=5]
  \tkzGrid \tkzClip
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(2,2){O}
  \tkzDefPoint(4,4){B}
  \tkzDefPoint(6,6){C}
  \tkzDrawPoints(O,A,B,C)
  \tkzLabelPoints(O,A,B,C)
  \tkzDrawCircle(O,A)
  \tkzClipCircle(O,A)
  \tkzDrawLine(A,C)
  \tkzDrawCircle[fill=red!20,opacity=.5](C,O)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

14.4.2 Exemple 2 de \tkzClipCircle



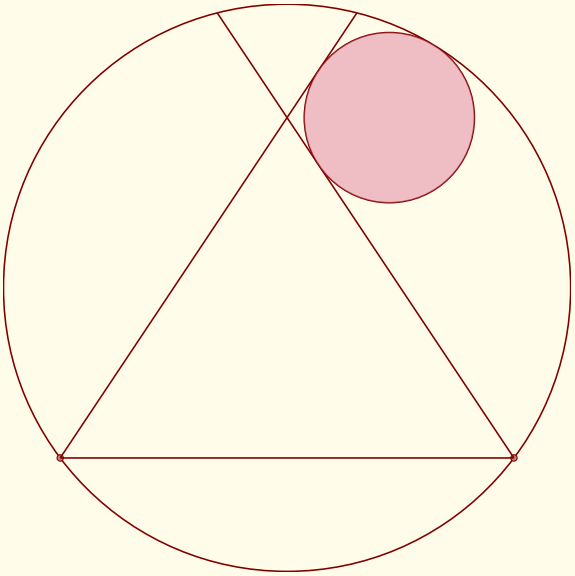

```
  \begin{tikzpicture}
  \tkzInit[xmax=6,ymax=6]
  \tkzGrid \tkzClip
   \tkzDefPoint(0,0){A}
   \tkzDefPoint(2,2){O}
   \tkzDefPoint(4,4){B}
   \tkzDefPoint(6,6){C}
   \tkzDrawPoints(O,A,B,C)
   \tkzLabelPoints(O,A,B,C)
   \tkzDrawCircle(O,A)
   \begin{scope}
    \tkzClipCircle(O,A)
    \tkzDrawLine(A,C)
   \end{scope}
   \tkzClipCircle[R](B,1cm)
  \tkzDrawCircle[fill=red!20,opacity=.5](C,B)
\end{tikzpicture}
```

14.4.3 Exemple 3 de \tkzClipCircle



```
\begin{tikzpicture}[scale=.5]
 \tkzDefPoint(0,0){A}
 \tkzDefPoint(8,0){B}
 \tkzDefSquare(A,B)\tkzGetPoints{C}{D}
 \tkzDrawPolygon(A,B,C,D)
 \tkzClipPolygon(A,B,C,D)
 \begin{scope}
  \tkzClipCircle(D,C)
  \tkzFillCircle[color=gray!50,%
                opacity=.5](B,A)
 \end{scope}
 \tkzDrawCircle(B,C)
 \tkzDrawCircle(D,C)
\end{tikzpicture}
```

14.4.4 Exemple 4 de \tkzClipCircle provenant d'un sangaku



```
\begin{tikzpicture}[scale=.75]
 \tkzInit[xmin=-5,ymin=-5,xmax=5,ymax=5]
 \tkzClip
 \tkzDefPoint(0,0){O}
 \tkzDefPoint(-2,-3){A}
 \tkzDefPoint(2,-3){B}
 \tkzDefPoint(0,3){Q}
 \tkzDrawCircle[R](O,5 cm)
 \tkzInterLC[R](A,B)(O,5 cm)
     \tkzGetPoints{M}{N}
 \tkzDrawPoints(M,N)
 \tkzClipCircle[R](O,5 cm)
 \tkzDrawLines[add= 1 and 1](A,B M,Q N,Q)
 \tkzDefMidPoint(M,N) \tkzGetPoint{R}
 \tkzDefLine[orthogonal=through Q](O,Q)
 \tkzGetPoint(q)
 \tkzCalcLength(R,Q) \tkzGetLength{dRQ}
 \tkzCalcLength(M,Q) \tkzGetLength{dMQ}
 \pgfmathparse{(\dMQ)/(\dRQ)*1.5}
 \edef\tkz@q{\pgfmathresult}%
 \tkzDefPoint(\tkz@q,3){K}
 \tkzDefPointBy[projection=onto N--Q](K)
    \tkzGetPoint{G}
  \tkzDrawCircle[R](K,1.5cm)
  \tkzFillCircle[R,color=purple!50,%
  opacity=.5](K,1.5 cm)
\end{tikzpicture}
```

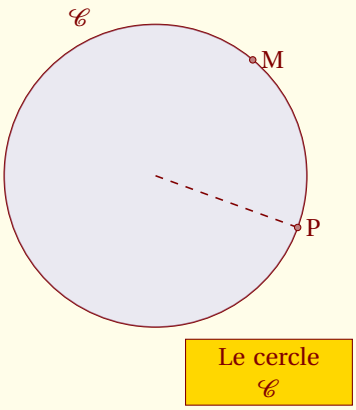
14.5 Donner un label à un cercle

\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* ***TikZ****. Le label est créé et donc "passé" entre accolades.*

14.5.1 Exemple de \tkzLabelCircle




```
\begin{tikzpicture}
  \tkzInit[ymin=-2.25,ymax=2.25,xmin=-2.25,xmax=2.25]
  \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\\ $\mathcal{C}$}
  \tkzDrawSegment[dashed](O,P)
  \tkzDrawPoints(M,P)\tkzLabelPoints[right](M,P)
\end{tikzpicture}
```

14.6 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. Ces macros remplacent d'anciennes macros qui existent encore **\tkzTgtFromP** ou **\tkzTgtFromPR** ainsi que **\tkzTgtAt**.

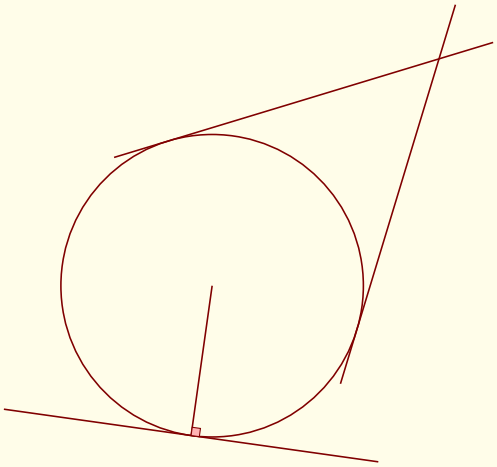
\tkzTangent[⟨*local options*⟩](⟨*pt1,pt2*⟩) ou (⟨*pt1,dim*⟩)

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

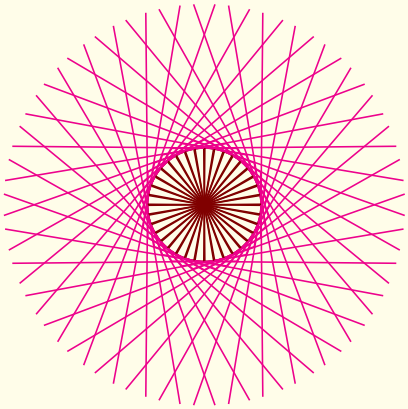
La tangente n'est pas tracée.Un second point de celle-ci est donné par ***tkzPointResult****.*

14.6.1 Exemple de tangente passant par un point du cercle



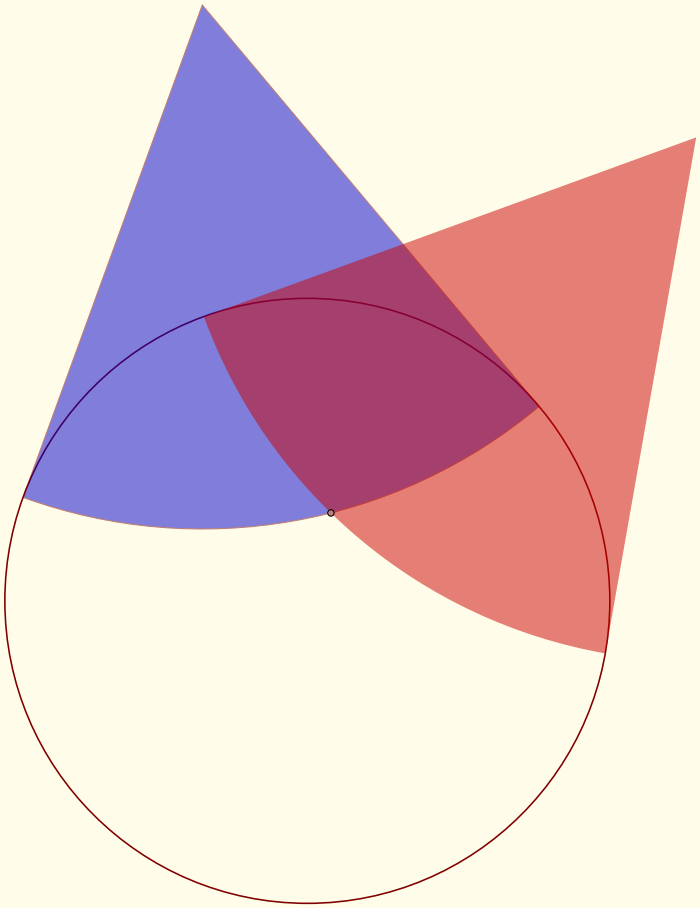
```
\begin{tikzpicture}[scale=.5]
  \tkzInit
  \tkzDefPoint(0,0){O}
   \tkzDefPoint(6,6){E}
  \tkzGetRandPointOn[circle=center O radius 4cm]{A}
  \tkzDrawSegment(O,A)
   \tkzDrawCircle(O,A)
  \tkzTangent[at=A](O)
   \tkzGetPoint{h}
  \tkzTangent[from=E](O,A)  \tkzGetPoints{e}{f}
    \tkzTangent[from with R=E](O,4 cm)
    \tkzGetPoints{k}{l}
    \tkzDrawLine[add = 5 and 4](A,h)
    \tkzMarkRightAngle[fill=red!30](O,A,h)
    \tkzDrawLines[](E,e E,l)
\end{tikzpicture}
```

14.6.2 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}
    \tkzTangent[from with R = a](c,\tkzRadius)
    \tkzGetPoints{e}{f}
    \tkzDrawLines[color=magenta](a,f a,e)
   \tkzDrawSegments(c,e c,f)}
\end{tikzpicture}
```

14.6.3 Exemple d'Andrew Mertz



```
\begin{tikzpicture}[scale=1]
  \tkzInit[xmin=-4.1,xmax=5.2,ymin=-4.1,ymax=8]
  \tkzClip[space=.5]
  \tkzDefPoint(100:8){A}\tkzDefPoint(50:8){B}
  \tkzDefPoint(0,0){C} \tkzDefPoint(0,4){R}
  \tkzDrawCircle(C,R)
  \tkzTangent[from = A](C,R)  \tkzGetPoints{D}{E}
  \tkzTangent[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/

SECTION 15

Utilisation du compas

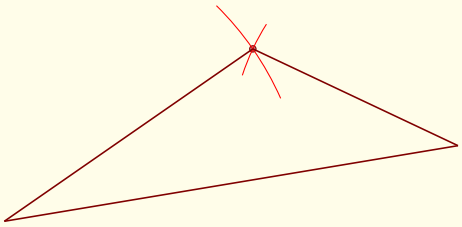
15.1 Macro principale \tkzCompass

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

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**, *soit en utilisant* **tkzPointResult** *s'il n'est pas nécessaire de conserver le nom.*

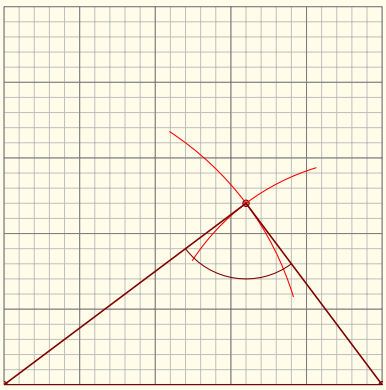
options	défaut	définition
delta	0	
length	0.75	
ratio	.5	

15.1.1 Option length



```
\begin{tikzpicture}
    \tkzInit[xmax=7,ymax=6]
    \tkzDefPoint[pos=left](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}
```

15.1.2 Option delta



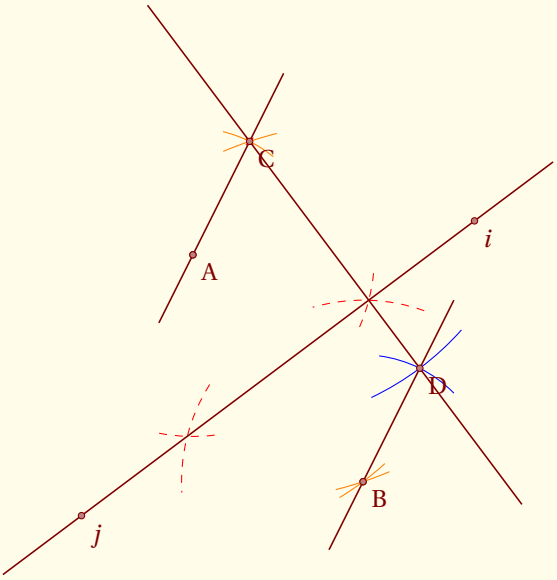
```
\begin{tikzpicture}
  \tkzInit[xmax=5,ymax=5]\tkzGrid[sub]
  \tkzClip
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(5,0){B}
  \tkzInterCC[R](A,4cm)(B,3cm)
  \tkzGetPoints{C}{D}
  \tkzDrawPoints(A,B,C)
  \tkzCompass[color=red,delta=20](A,C)
  \tkzCompass[color=red,delta=20](B,C)
  \tkzDrawPolygon(A,B,C)
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

15.2 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	
length	0.75	
ratio	.5	

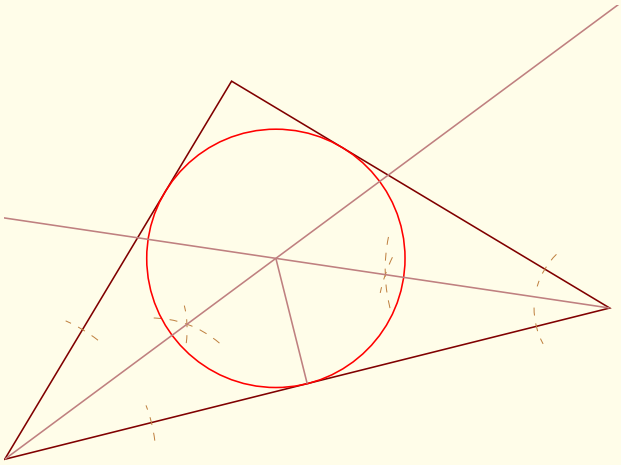



```
\begin{tikzpicture}[scale=.75]
 \tkzDefPoint(2,2){A}  \tkzDefPoint(5,-2){B}
 \tkzDefPoint(3,4){C}  \tkzDrawPoints(A,B)
 \tkzDrawPoint[color=red,shape=cross out](C)
 \tkzCompasss[color  = orange,length = 1](A,B A,C B,C C,B)
 \tkzShowLine[mediator,color=red,dashed,length = 2](A,B)
 \tkzShowLine[parallel = 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}
```

15.3 Macro de configuration \tkzSetUpCompass

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

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



```
\begin{tikzpicture}
  \tkzInit[xmax=9,ymax=7] \tkzClip
  \tkzDefPoints{0/1/A, 8/3/B, 3/6/C}
  \tkzDrawPolygon(A,B,C)
  \tkzSetUpCompass[color=brown,line width=.3 pt,style=dashed]
  \tkzDefLine[bisector](B,A,C)  \tkzGetPoint{a}
  \tkzDefLine[bisector](C,B,A)  \tkzGetPoint{b}
  \tkzShowLine[bisector,size=2,gap=3](B,A,C)
  \tkzShowLine[bisector,size=1,gap=3](C,B,A)
  \tkzInterLL(A,a)(B,b) \tkzGetPoint{I}
  \tkzDefPointBy[projection= onto A--B](I) \tkzGetPoint{H}
  \tkzDrawCircle[radius,color=red](I,H)
  \tkzDrawSegments[color=Maroon!50](I,H)
  \tkzDrawLines[add=0 and 5,color=Maroon!50 ](A,a B,b)
\end{tikzpicture}
```

SECTION 16

Les secteurs

\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

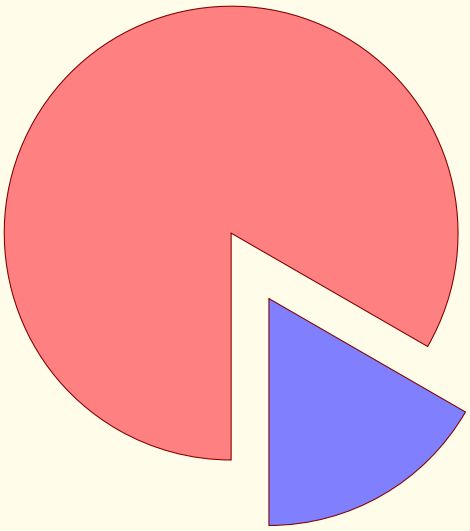
Il faut ajouter bien sûr tous les styles de ***TikZ*** *pour les tracés*

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 :

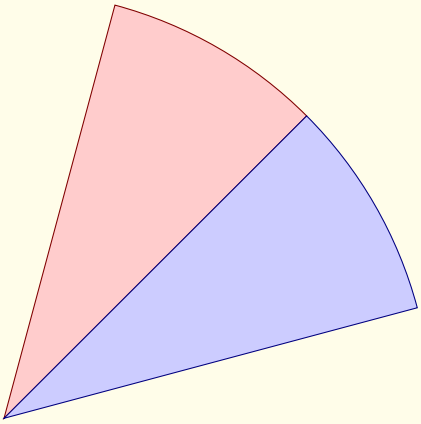
16.1 \tkzDrawSector et towards

Il est inutile de mettre **towards**.



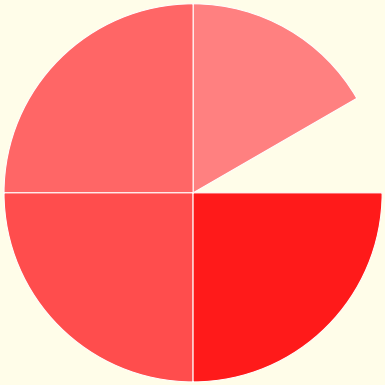
```
\begin{tikzpicture}[scale=1]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(-30:3){A}
  \tkzDefPointBy[rotation = center O angle -60](A)
  \tkzDrawSector[fill=red!50](O,A)(tkzPointResult)
 \begin{scope}[shift={(-60:1cm)}]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(-30:3){A}
  \tkzDefPointBy[rotation = center O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

16.2 \tkzDrawSector et rotate



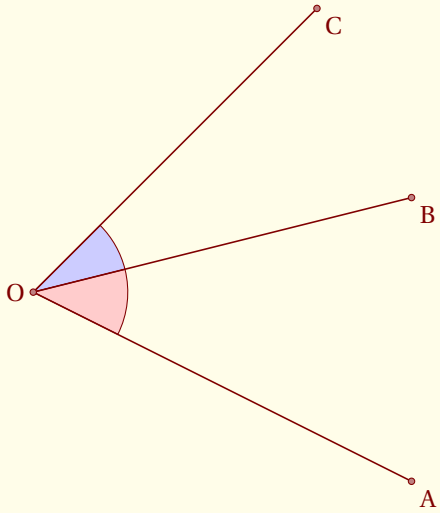
```
\begin{tikzpicture}[scale=2]
   \tkzDefPoint(0,0){O}
   \tkzDefPoint(2,2){A}
   \tkzDrawSector[rotate,draw=red!50!black,%
   fill=red!20](O,A)(30)
   \tkzDrawSector[rotate,draw=blue!50!black,%
   fill=blue!20](O,A)(-30)
\end{tikzpicture}
```

16.3 \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}
```

16.4 \tkzDrawSector et R




```
\begin{tikzpicture}[scale=1.25]
 \tkzDefPoint[pos=left](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 cm)(A,B)
\tkzDrawSegments(O,A O,B O,C)
\tkzDrawPoints(O,A,B,C)
\tkzLabelPoints(A,B,C)
\tkzLabelPoints[left](O)
\end{tikzpicture}
```

\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 **TikZ** *pour 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)**

16.5 \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}
```

16.6 \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}
```

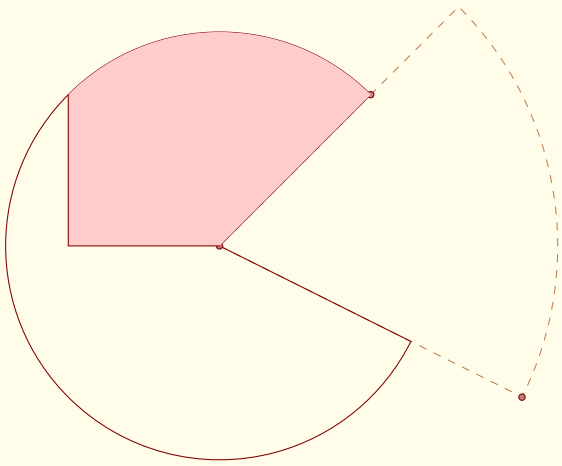
\tkzClipSector[⟨*local options*⟩](⟨*O,...*⟩)(⟨*...*⟩)

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 ***TikZ*** *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)**



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

SECTION 17

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

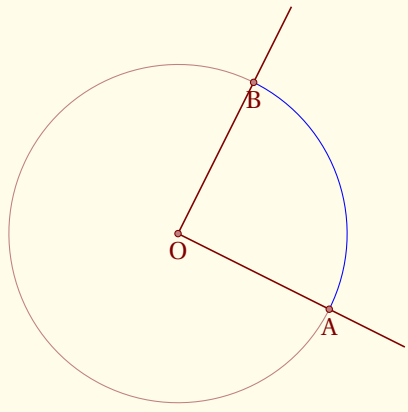
Il faut ajouter bien sûr tous les styles de **TikZ** *pour les tracés*

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 :

17.1 \tkzDrawArc et towards

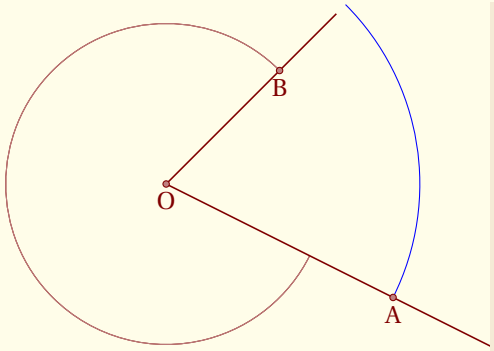
Il est inutile de mettre **towards**. Dans ce premier exemple l'arc part de A et va sur B. L'arc qui va de B vers A est différent. On obtient le saillant en allant dans le sens direct du cercle trigonométrique.




```
\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}
```

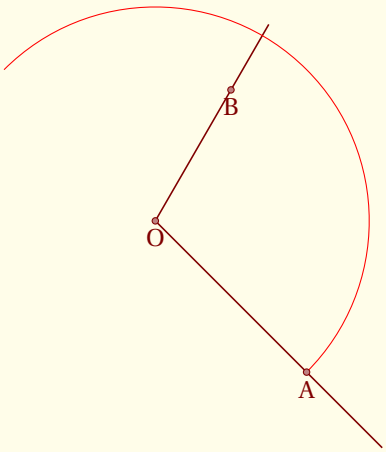
17.2 \tkzDrawArc et towards

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



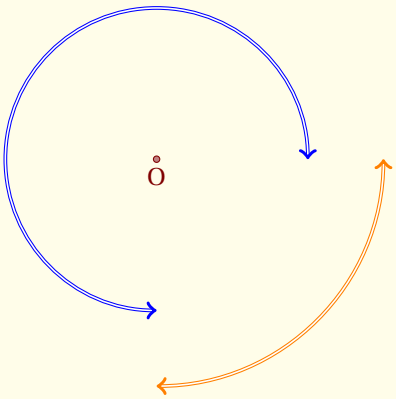

```
\begin{tikzpicture}[scale=1.5]
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2,-1){A}
  \tkzDefPoint(1,1){B}
  \tkzDrawArc[color=blue](O,A)(B)
  \tkzDrawArc[color=Maroon](O,B)(A)
  \tkzDrawArc(O,B)(A)
  \tkzDrawLines[add = 0 and .5](O,A O,B)
  \tkzDrawPoints(O,A,B)
  \tkzLabelPoints[below](O,A,B)
\end{tikzpicture}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

17.3 \tkzDrawArc et rotate



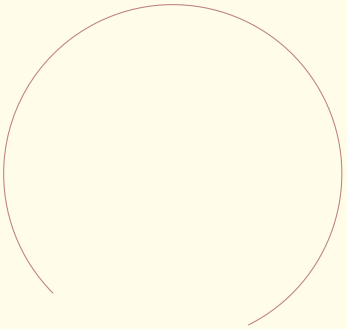

```
\begin{tikzpicture}
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2,-2){A}
  \tkzDefPoint(60:2){B}
  \tkzDrawLines[add = 0 and .5](O,A O,B)
  \tkzDrawArc[rotate,color=red](O,A)(180)
  \tkzDrawPoints(O,A,B)
  \tkzLabelPoints[below](O,A,B)
\end{tikzpicture}
```

17.4 \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}
```

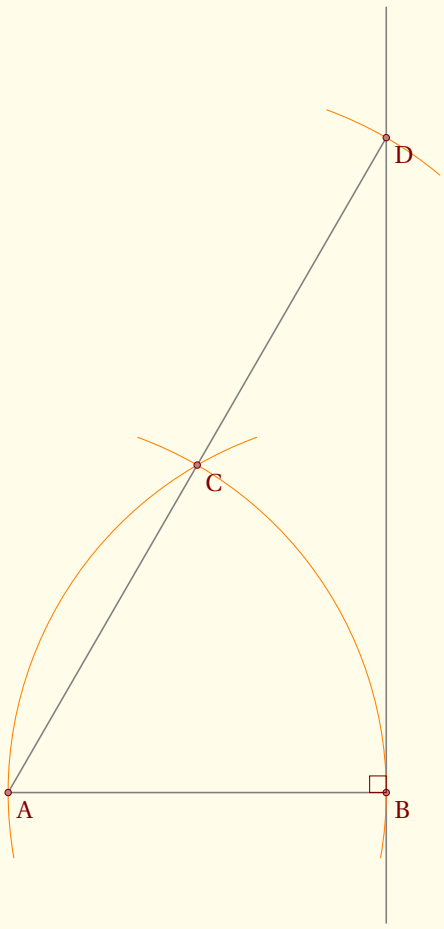
17.5 \tkzDrawArc et R with nodes



```
\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}
```

17.6 \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}
```

SECTION 18

Rapporteurs

D'après une idée de Yves Combe., la macro suivante permet de dessiner un rapporteur. J'ai ajouté mon propre rapporteur qui est obtenu avec l'option **full** (par défaut), celui de Yves est obtenu avec **half**.

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

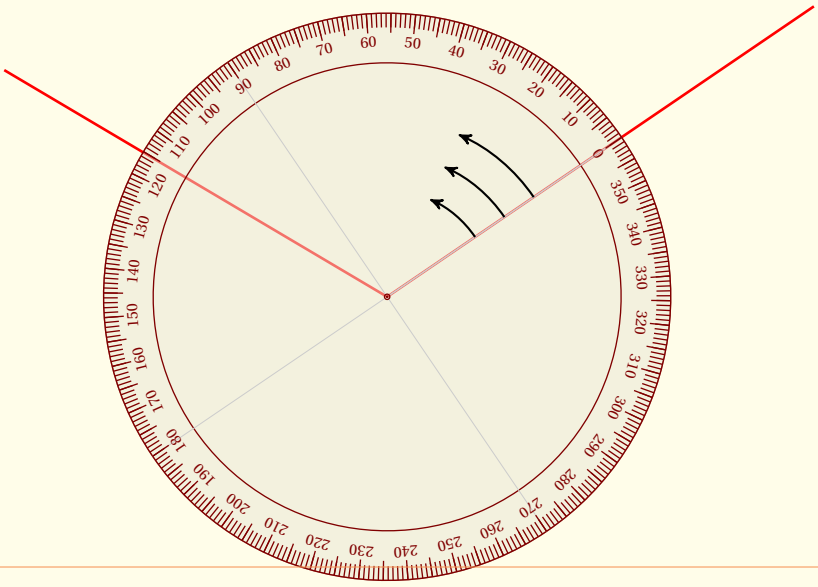
options	défaut	définition
with	full	full ou bien half
lw	0.4 pt	épaisseur des lignes
scale	1	ratio : permet d'ajuster la taille du rapporteur
return	false	sens indirect du cercle trigonométrique

Le principe de fonctionnement est encore plus simple. Il suffit de nommer une demi-droite. Le rapporteur sera placé sur l'origine O *la direction de la demi-droites est donnée par* A. *L'angle est mesuré dans le sens direct du cercle trigonométrique*

18.1 Le rapporteur circulaire

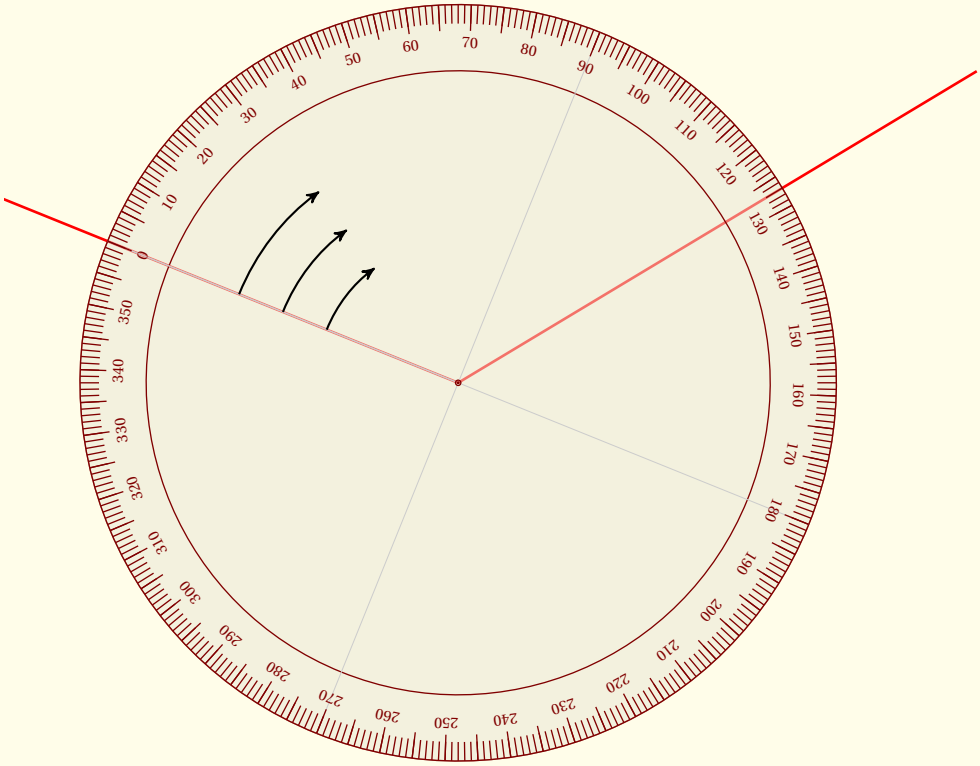
Mesure dans le sens direct

```
\begin{tikzpicture}[scale=.75]
\tkzDefPoint(2,3){A}
\tkzDefPoint[shift={(2,3)}](31:8){B}
\tkzDefPoint[shift={(2,3)}](158:8){C}
\tkzDrawSegments[color = red,
          line width = 1pt](A,B A,C)
\tkzProtractor[with  = full,
               scale = 1.25](A,B)
\end{tikzpicture}
```



18.2 Le rapporteur circulaire, transparent et retourné

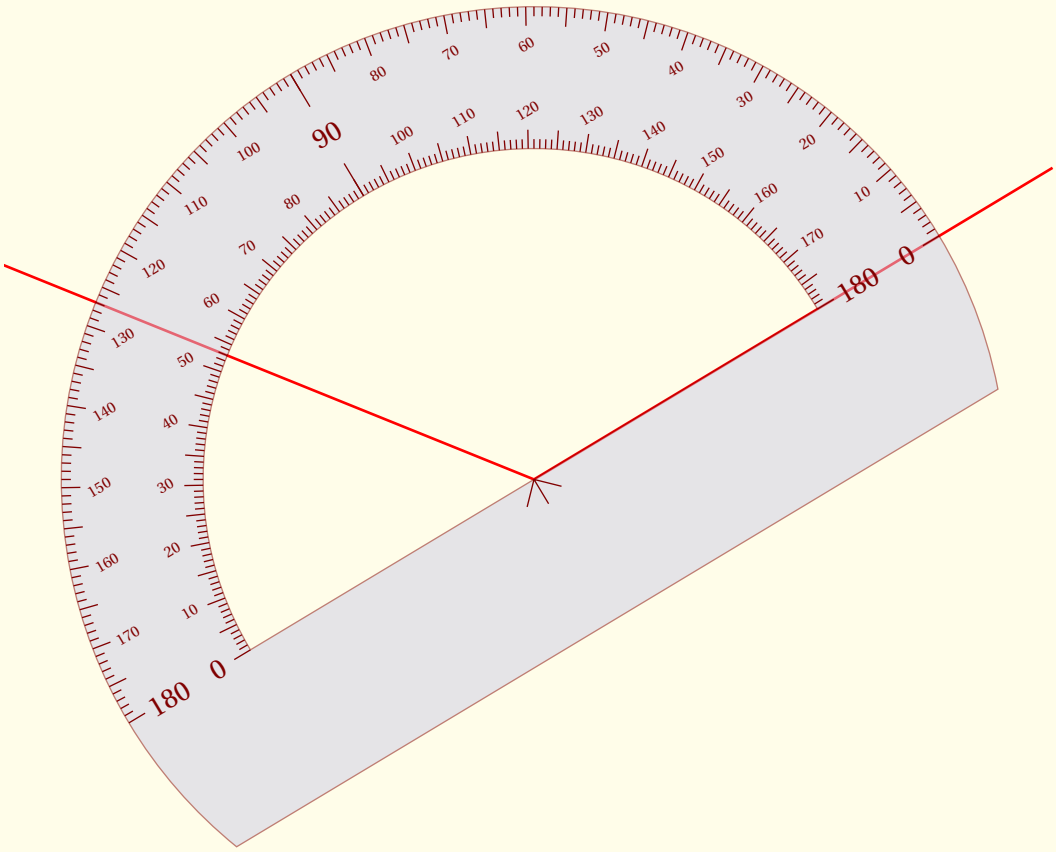
Mesure dans le sens indirect, on retourne le rapporteur.




```
\begin{tikzpicture}
  \tkzInit[xmin=-4,xmax=9,ymin=-3,ymax=9]
  \tkzClip
  \tkzDefPoint(2,3){A}
  \tkzDefPoint[shift={(2,3)}](31:8){B}
  \tkzDefPoint[shift={(2,3)}](158:8){C}
  \tkzDrawSegments[color=red,line width=1pt](A,B A,C)
  \tkzProtractor[scale=1.25,with=full,return](A,C)
\end{tikzpicture}
```

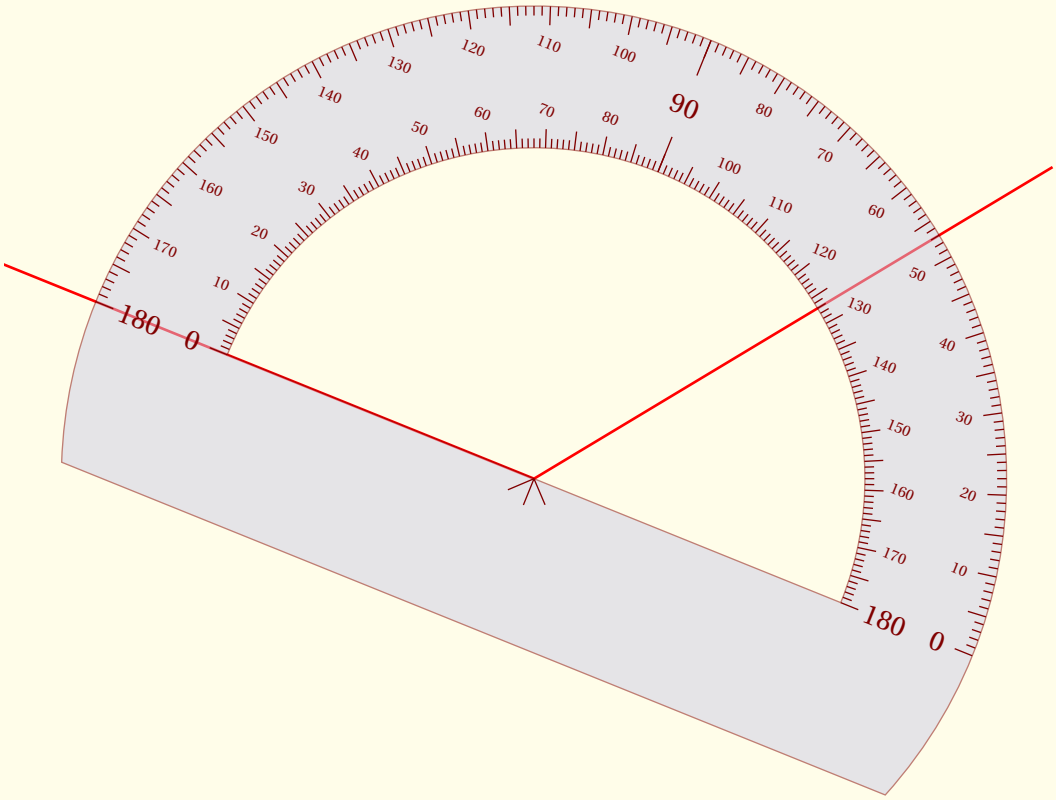
18.3 Le rapporteur original semi-circulaire (Yves Combes)

Mesure dans le sens direct avec un rapporteur semi-circulaire




```
\begin{tikzpicture}
  \tkzInit[xmin=-5,xmax=9,ymin=-3,ymax=10]
  \tkzClip
  \tkzDefPoint(2,3){A}
  \tkzDefPoint[shift={(2,3)}](31:8){B}
  \tkzDefPoint[shift={(2,3)}](158:8){C}
  \tkzDrawSegments[color=red,line width=1pt](A,B A,C)
  \tkzProtractor[scale=1.25,with=half](A,B)
\end{tikzpicture}
```

18.4 Le rapporteur semi-circulaire dans le sens indirect



```
\begin{tikzpicture}
  \tkzInit[xmin=-5,xmax=9,ymin=-3,ymax=10]
  \tkzClip
  \tkzDefPoint(2,3){A}
  \tkzDefPoint[shift={(2,3)}](31:8){B}
  \tkzDefPoint[shift={(2,3)}](158:8){C}
  \tkzDrawSegments[color=red,line width=1pt](A,B A,C)
  \tkzProtractor[scale=1.25,with=half,return](A,C)
\end{tikzpicture}
```

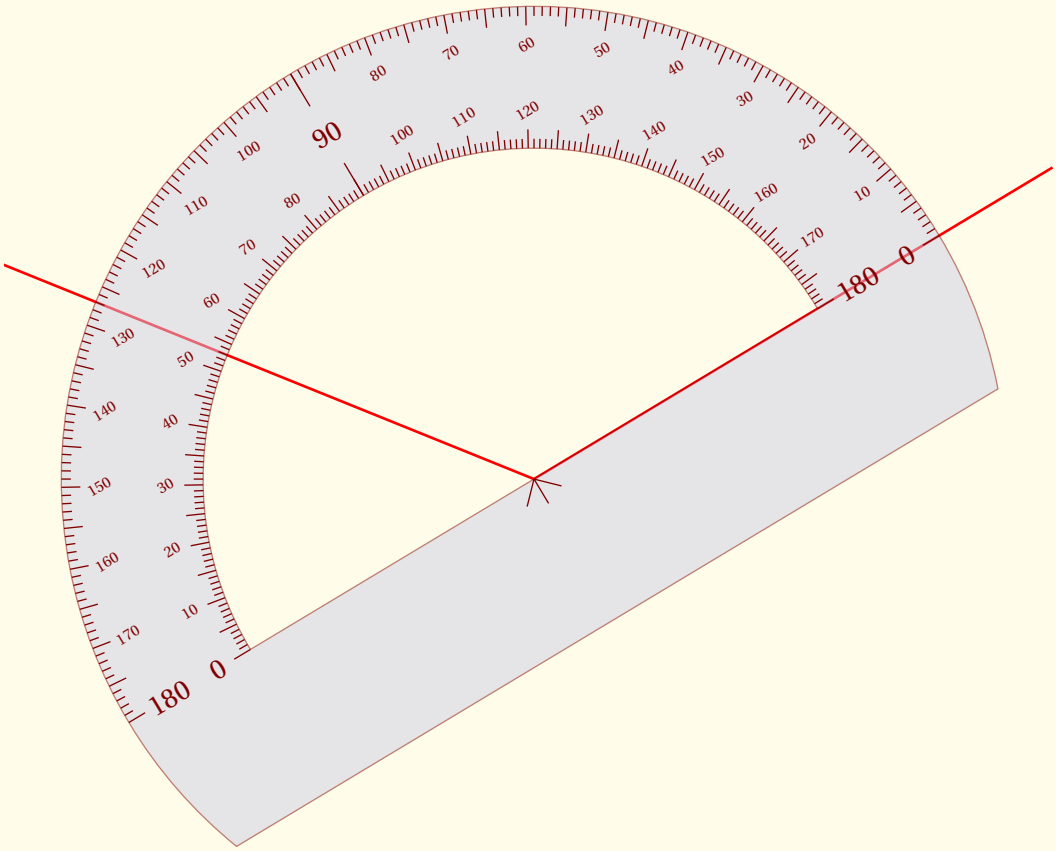
le cas échéant vous pouvez utiliser la macro originale de Yves

\tkzOriProtractor[⟨*local options*⟩]

options	défaut	définition
with	full	full ou bien half
lw	0.4 pt	épaisseur des lignes
shift	(x;y)	permet de faire glisser le rapporteur
rotate	0	permet de faire pivoter le rapporteur
scale	1	ratio : permet d'ajuster la taille du rapporteur
return	false	sens indirect du cercle trigonométrique

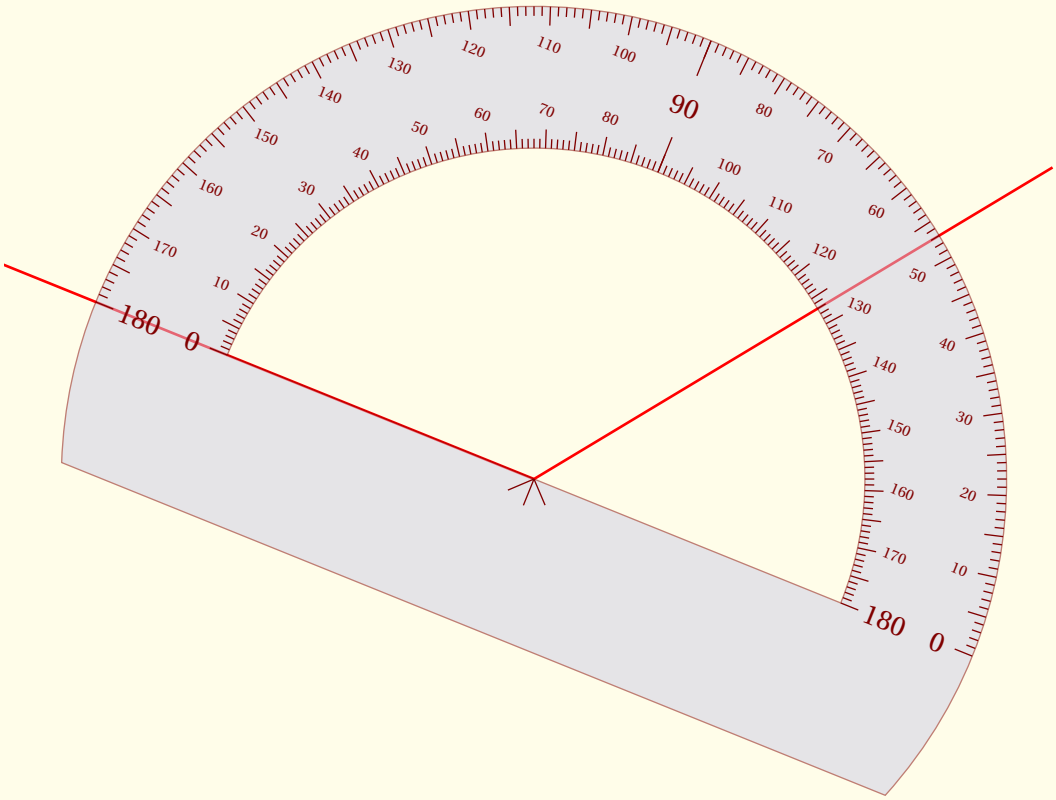
Le principe de fonctionnement est encore plus simple. Il suffit de nommer une demi-droite. Le rapporteur sera placé sur l'origine.

18.5 Le rapporteur semi-circulaire avec la macro originale



```
\begin{tikzpicture}
  \tkzInit[xmin=-5,xmax=9,ymin=-3,ymax=10]
  \tkzClip
  \tkzDefPoint(2,3){A}
  \tkzDefPoint[shift={(2,3)}](158:8){B}
  \tkzDefPoint[shift={(2,3)}](31:8){C}
  \tkzDrawSegments[color=red,line width=1pt](A,B A,C)
  \tkzOriProtractor[shift = {(2,3)},scale=1.25, rotate = +31,with=half]
\end{tikzpicture}
```

18.6 Le rapporteur semi-circulaire avec la macro originale dans le sens indirect




```
\begin{tikzpicture}
  \tkzInit[xmin=-5,xmax=9,ymin=-3,ymax=10]
  \tkzClip
  \tkzDefPoint(2,3){A}
  \tkzDefPoint[shift={(2,3)}](158:8){B}
  \tkzDefPoint[shift={(2,3)}](31:8){C}
  \tkzDrawSegments[color=red,line width=1pt](A,B A,C)
  \tkzOriProtractor[shift = {(2,3)},scale=1.25, rotate = -22,with=half]
\end{tikzpicture}
```

SECTION 19

Quelques outils

19.1 Dupliquer un segment

Il s'agit de construire un segment sur une demi-droite donnée de même longueur qu'un segment donné.

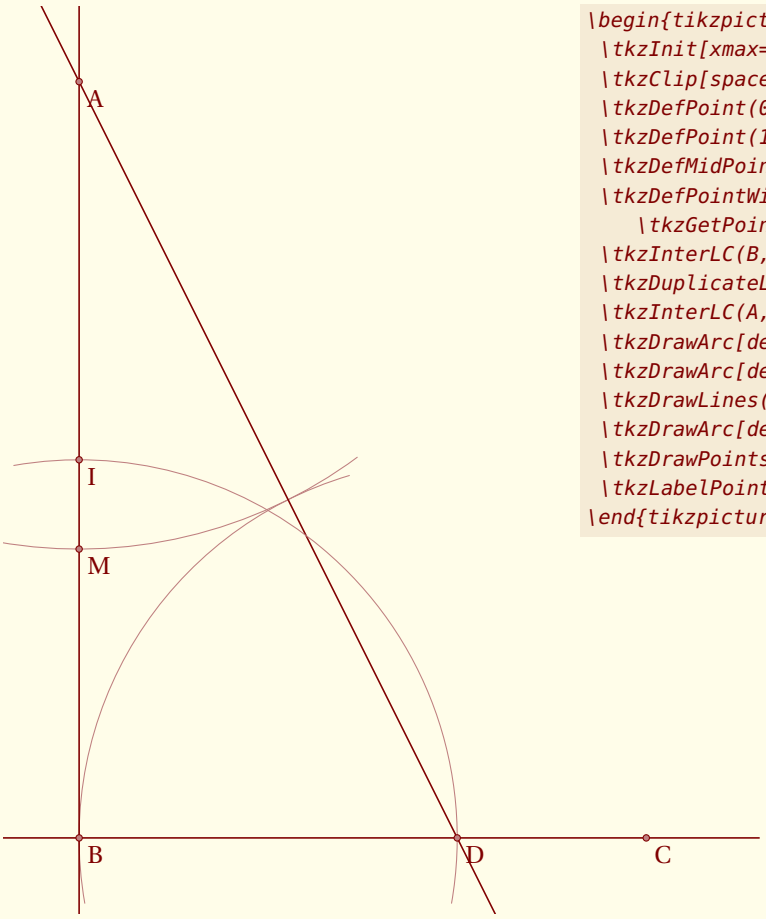
\tkzDuplicateLen(⟨*pt1,pt2*⟩)(⟨*pt3,pt4*⟩){⟨*pt5*⟩}

Il s'agit de créer un segment sur une demi-droite donnée de même longueur qu'un segment donné . Il s'agit en fait de la définition d'un point.

arguments	exemple	explication
(pt1,pt2)(pt3,pt4){pt5}	**\tkzDuplicateLen**(A,B)(E,F){C}	AC=EF et C ∈ [AB)

La macro ***\tkzDuplicateSegment*** *est identique à celle-ci.*

19.1.1 Proportion d'or avec \tkzDuplicateLen




```
\begin{tikzpicture}[rotate=-90]
 \tkzInit[xmax=10,ymax=10]
 \tkzClip[space=1]
 \tkzDefPoint(0,0){A}
 \tkzDefPoint(10,0){B}
 \tkzDefMidPoint(A,B)       \tkzGetPoint{I}
 \tkzDefPointWith[orthogonal,K=-.75](B,A)
    \tkzGetPoint{C}
 \tkzInterLC(B,C)(B,I)      \tkzGetSecondPoint{D}
 \tkzDuplicateLen(B,D)(D,A) \tkzGetPoint{E}
 \tkzInterLC(A,B)(A,E)      \tkzGetPoints{N}{M}
 \tkzDrawArc[delta=10](D,E)(B)
 \tkzDrawArc[delta=10](A,M)(E)
 \tkzDrawLines(A,B B,C A,D)
 \tkzDrawArc[delta=10](B,D)(I)
 \tkzDrawPoints(A,B,D,C,M,I,N)
 \tkzLabelPoints(A,B,D,C,M,I,N)
\end{tikzpicture}
```

19.2 Déterminer une pente

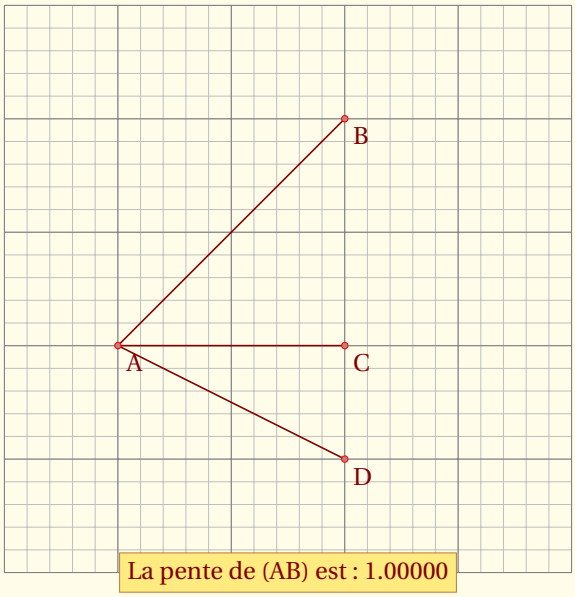
Il s'agit de déterminer si elle existe, la pente d'une droite définie par deux points. Aucune vérification de l'existence n'est faite.

\tkzFindSlope(⟨*pt1,pt2*⟩){⟨*name of macro*⟩}

Le résultat est stocké dans une macro.

arguments	exemple	explication
(pt1,pt2)pt3	**\tkzFindSlope**(A,B){slope}	**\slope** donnera le résultat de $\frac{y_B-y_A}{x_B-x_A}$

Attention à ne pas avoir $x_B = x_A$



La pente de (AB) est : 1.00000

La pente de (AC) est : 0.00000

La pente de (AD) est : -0.49999

```
\begin{tikzpicture}[scale=1.5]
  \tkzInit[xmax=5,ymax=5]\tkzGrid[sub]
  \tkzDefPoint(1,2){A}    \tkzDefPoint(3,4){B}
  \tkzDefPoint(3,2){C}    \tkzDefPoint(3,1){D}
  \tkzDrawSegments(A,B A,C A,D)
  \tkzDrawPoints[color=red](A,B,C,D)  \tkzLabelPoints(A,B,C,D)
  \tkzFindSlope(A,B){SAB} \tkzFindSlope(A,C){SAC}\tkzFindSlope(A,D){SAD}
  \tkzText[fill=Gold!50,draw=brown](2.5,0){La pente de (AB) est : \SAB}
  \tkzText[fill=Gold!50,draw=brown](2.5,-.5){La pente de (AC) est : \SAC}
  \tkzText[fill=Gold!50,draw=brown](2.5,-1){La pente de (AD) est : \SAD}
\end{tikzpicture}
```

19.3 Angle formé par une droite avec l'axe horizontal

Beaucoup plus intéressante que la précédente. Le résultat est compris entre -180 degrés et +180 degrés.

\tkzFindSlopeAngle(⟨*pt1,pt2*⟩)

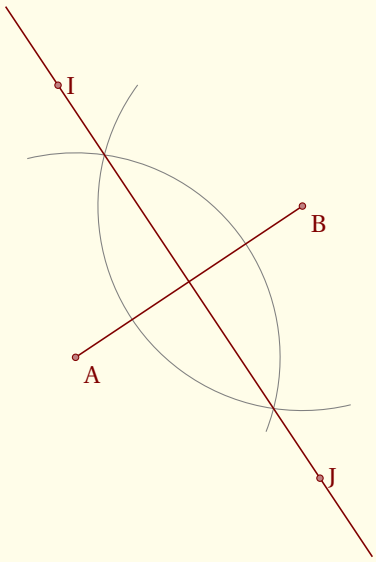
Le résultat est stocké dans une macro **\tkzAngleResult**.

arguments	exemple	explication
(pt1,pt2)	**\tkzFindSlopeAngle**(A,B)	**\tkzGetAngle** peut récupèrer le résultat

Si la récupération n'est pas nécessaire, il est possible d'utiliser ***\tkzAngleResult***

19.3.1 exemple d'utilisation de \tkzFindSlopeAngle

Voici une autre version de la construction d'une médiatrice




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

19.4 Récupérer un angle

Dans l'exemple précédent, j'ai utilisé la macro **\tkzGetAngle** qui permet de récupérer un angle.

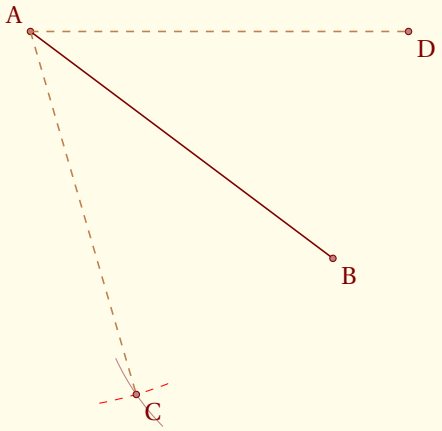
\tkzGetAngle{⟨*name of macro*⟩}

Cette macro récupère **\tkzAngleResult** et stocke le résultat dans une nouvelle macro.

arguments	exemple	explication
name of macro	**\tkzGetAngle**{ang}	**\ang** contient la valeur de l'angle.

19.5 exemple d'utilisation de \tkzGetAngle

Il s'agit ici que (AB) soit la bissectrice de $\widehat{CAD}$, tel que la pente AD soit nulle. On récupère la pente de (AB) puis on effectue deux rotations.




```
\begin{tikzpicture}
  \tkzInit
  \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,dashed,color=red](A,C)
  \tkzCompass[delta=10,Maroon](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}
```

19.6 Angle formé par trois points

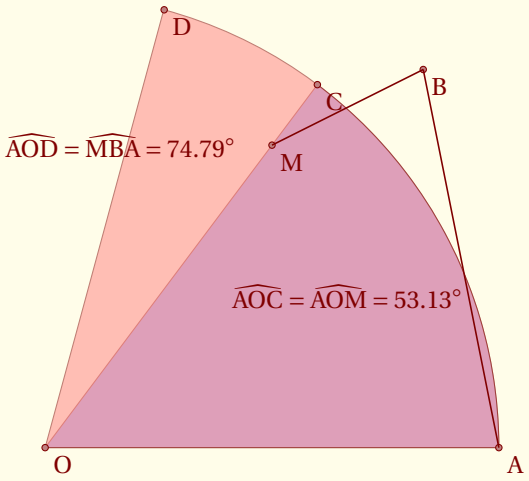
\tkzFindAngle(⟨*pt1,pt2,pt3*⟩)

Le résultat est stocké dans une macro **\tkzAngleResult**.

arguments	exemple	explication
(pt1,pt2,pt3)	**\tkzFindAngle**(A,B,C)	**\tkzAngleResult** donne l'angle $(\overrightarrow{BA},\overrightarrow{BC})$

Le résultat est compris entre -180 degrés et +180 degrés. pt2 est le sommet et ***\tkzGetAngle*** *peut récupérer l'angle.*

19.7 Exemple d'utilisation de \tkzFindAngle




```
\begin{tikzpicture}
  \tkzInit[xmin=-1,ymin=-1,xmax=7,ymax=7]
  \tkzClip
  \tkzDefPoint (0,0){O}  \tkzDefPoint (6,0){A}
  \tkzDefPoint (5,5){B}  \tkzDefPoint (3,4){M}
  \tkzFindAngle (A,O,M)  \tkzGetAngle{an}
  \tkzDefPointBy[rotation=center O angle \an](A) \tkzGetPoint{C}
  \tkzDrawSector[fill = blue!50,opacity=.5](O,A)(C)
  \tkzFindAngle(M,B,A)   \tkzGetAngle{am}
  \tkzDefPointBy[rotation = center O angle \am](A) \tkzGetPoint{D}
  \tkzDrawSector[fill = red!50,opacity = .5](O,A)(D)
  \tkzDrawPoints(O,A,B,M,C,D)   \tkzLabelPoints(O,A,B,M,C,D)
  \FPround\an\an{2} \FPround\am\am{2} \tkzDrawSegments(M,B B,A)
  \tkzText(4,2){$\widehat{AOC}=\widehat{AOM}=\an^{\circ}$}
  \tkzText(1,4){$\widehat{AOD}=\widehat{MBA}=\am^{\circ}$}
\end{tikzpicture}
```

19.8 Longueur d'un segment \tkzVecLen

Il existe dans **TikZ** une option **veclen**. Cette option permet de calculer AB si A et B sont deux points.

Le seul problème pour moi est que la version de **TikZ** n'est pas assez précise dans certains cas particuliers. Ma version utilise le package **fp.sty** et est plus lente, mais plus précise

\tkzVecLen[⟨*local options*⟩**](**⟨*pt1,pt2*⟩**){**⟨*name of macro*⟩**}**

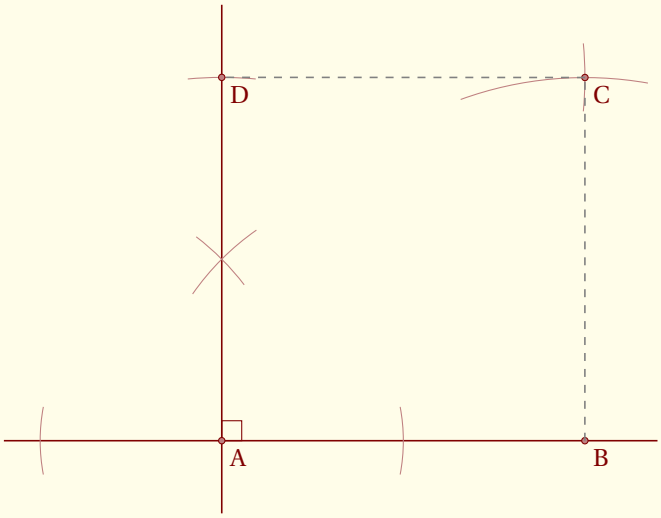
Le résultat est stocké dans une macro.

arguments	exemple	explication
(pt1,pt2){name of macro}	**\tkzVecLen**(A,B){dAB}	**\dAB** donne AB en pt

Une seule option

options	défaut	exemple
cm	false	**\tkzVecLen**[cm](A,B){dAB} **\dAB** donne AB en cm

19.8.1 Construction d'un carré au compas




```
\begin{tikzpicture}[scale=1.2]
  \tkzDefPoint(0,0){A} \tkzDefPoint(4,0){B}
  \tkzDrawLine[add= .6 and .2](A,B)
  \tkzCalcLength[cm](A,B)\tkzGetLength{dAB}
  \tkzDefLine[perpendicular=through A](A,B)
  \tkzDrawLine(A,tkzPointResult) \tkzGetPoint{D}
  \tkzShowLine[orthogonal=through A,gap=2](A,B)
  \tkzMarkRightAngle(B,A,D)
  \tkzVecKOrth[-1](B,A){C}
  \tkzCompasss(A,D D,C)   \tkzDrawArc[R](B,\dAB)(80,110)
  \tkzDrawPoints(A,B,C,D) \tkzDrawSegments[color=gray,style=dashed](B,C C,D)
  \tkzLabelPoints(A,B,C,D)
\end{tikzpicture}
```

19.9 Transformation de pt en cm ou de cm en pt

Pas sûr que cela soit nécessaire et il ne s'agit que d'une division par 28,45274 et d'un multiplication par ce même nombre. Les macros sont :

\tkzpttocm(⟨*nombre*⟩){⟨*name of macro*⟩}

Le résultat est stocké dans une macro.

arguments	exemple	explication
(nombre)name of macro	**\tkzpttocm**(120){len}	**\len** donne un nombre de tkznamecm

Il faudra utiliser **\len** *accompagné de* **cm**

\tkzcmtopt(⟨*nombre*⟩){⟨*name of macro*⟩}

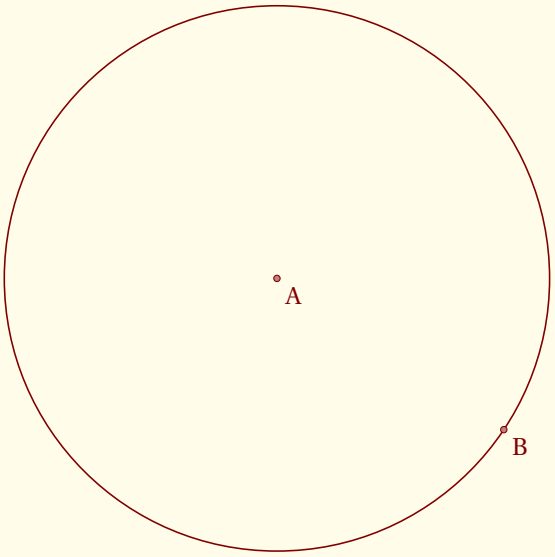
Le résultat est stocké dans une macro.

arguments	exemple	explication
(nombre)name of macro	**\tkzcmtopt**(5){len}	**\len** donne un nombre de tkznamept

Il faudra utiliser **\len** *accompagné de* **pt**

19.9.1 Exemple

La macro **\tkzDefCircle[radius](A,B)** définit le rayon que l'on récupère avec **\tkzGetLength**, mais ce résultat est en **pt**.




```
\begin{tikzpicture}
   \tkzDefPoint(0,4){A}
   \tkzDefPoint(3,2){B}
   \tkzDefCircle[radius](A,B)
   \tkzGetLength{rABpt}
   \tkzpttocm(\rABpt){rABcm}
   \tkzDrawCircle(A,B)
   \tkzDrawPoints(A,B)
   \tkzLabelPoints(A,B)
\end{tikzpicture}
```

SECTION 20

Personnalisation

20.1 Fichier de configuration : tkz-base.cfg

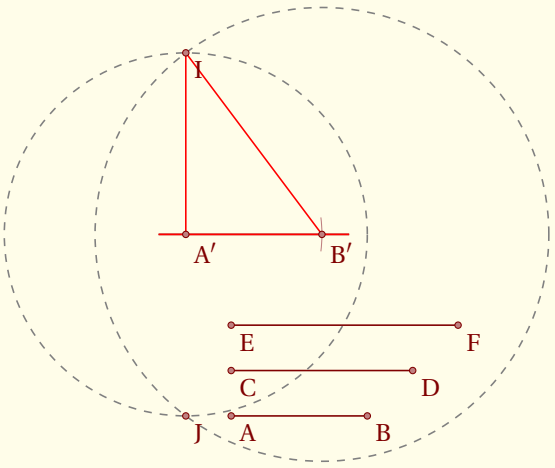
Vous pouvez créer votre propre fichier **tkz-base.cfg** que vous placerez dans un dossier qui sera prioritaire au sein du **texmf**. Dans **tkz-base.cfg**, il est possible de modifier les couleurs, ls épaisseurs des lignes. La lecture de ce fichier doit suffire à déterminer le rôle de chaque variable.

20.2 \tkzSetUpLine

\tkzSetUpLine[⟨*local options*⟩]

options	défaut	définition
color	black	couleur des arcs de cercle de construction
line width	0.4pt	épaisseur des arcs de cercle de construction
style	solid	style des arcs de cercle de construction
add	.2 and .2	modification de la longueur d'un segment

Construire un triangle avec trois segments donnés




```
\begin{tikzpicture}[scale=.6]
 \tkzDefPoint(1,0){A} \tkzDefPoint(4,0){B}
 \tkzDefPoint(1,1){C} \tkzDefPoint(5,1){D}
 \tkzDefPoint(1,2){E} \tkzDefPoint(6,2){F}
 \tkzDefPoint(0,4){A'}\tkzDefPoint(3,4){B'}
 \tkzDrawSegments(A,B C,D E,F)
 \tkzDrawLine(A',B')
 \tkzSetUpLine[style=dashed,color=gray]
 \tkzCompass(A',B')
 \tkzCalcLength[cm](C,D)  \tkzGetLength{rCD}
 \tkzDrawCircle[R](A',\rCD cm)
 \tkzCalcLength[cm](E,F)  \tkzGetLength{rEF}
 \tkzDrawCircle[R](B',\rEF cm)
 \tkzInterCC[R](A',\rCD cm)(B',\rEF cm)
 \tkzGetPoints{I}{J}
 \tkzSetUpLine[color=red] \tkzDrawLine(A',B')
 \tkzDrawSegments(A',I B',I)
 \tkzDrawPoints(A,B,C,D,E,F,A',B',I,J)
 \tkzLabelPoints(A,B,C,D,E,F,A',B',I,J)
\end{tikzpicture}
```

Par défaut, dans **tkz-base.cfg**, ces styles sont définis par :

```
\global\edef\tkz@euc@linecolor{\tkz@mainlinecolor}
\global\def\tkz@euc@linewidth{0.6pt}
\global\def\tkz@euc@linestyle{solid}
\global\def\tkz@euc@lineleft{.2}
\global\def\tkz@euc@lineright{.2}
```

20.3 \tkzSetUpCompass

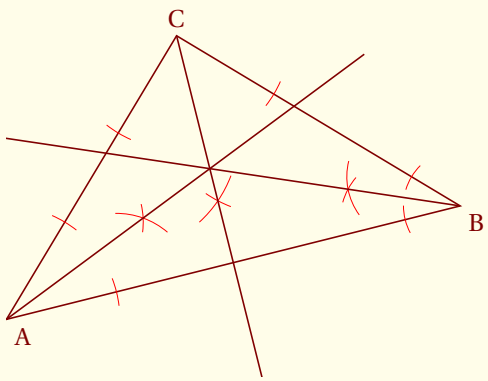
\tkzSetUpCompass[⟨*local options*⟩]

options	défaut	définition
color	black	couleur des arcs de cercle de construction
line width	0.4pt	épaisseur des arcs de cercle de construction
style	solid	style des arcs de cercle de construction

Par défaut, dans **tkz-base.cfg**, ces styles sont définis par :

```
\global\edef\tkz@euc@compasscolor{\tkz@otherlinecolor}
\global\def\tkz@euc@compasswidth{0.4pt}
\global\def\tkz@euc@compassstyle{solid}
```

Vous pouvez créer votre propre fichier **tkz-base.cfg** que vous placerez dans un dossier qui sera prioritaire au sein du **texmf**.




```
\begin{tikzpicture}[scale=0.75]
  \tkzInit[ymax=8] \tkzClip
  \tkzDefPoints{0/1/A, 8/3/B, 3/6/C}
  \tkzDrawPolygon(A,B,C)
  \tkzSetUpCompass[color=red,line width=.2 pt]
  \tkzDefLine[bisector](A,C,B) \tkzGetPoint{c}
  \tkzDefLine[bisector](B,A,C) \tkzGetPoint{a}
  \tkzDefLine[bisector](C,B,A) \tkzGetPoint{b}
  \tkzShowLine[bisector,size=2,gap=3](A,C,B)
  \tkzShowLine[bisector,size=2,gap=3](B,A,C)
  \tkzShowLine[bisector,size=1,gap=2](C,B,A)
  \tkzDrawLines[add=0 and 0 ](B,b C,c)
  \tkzDrawLine[add=0 and -.4 ](A,a)
  \tkzLabelPoints(A,B) \tkzLabelPoints[above](C)
\end{tikzpicture}
```

SECTION 21

Quelques exemples intéressants

21.1 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

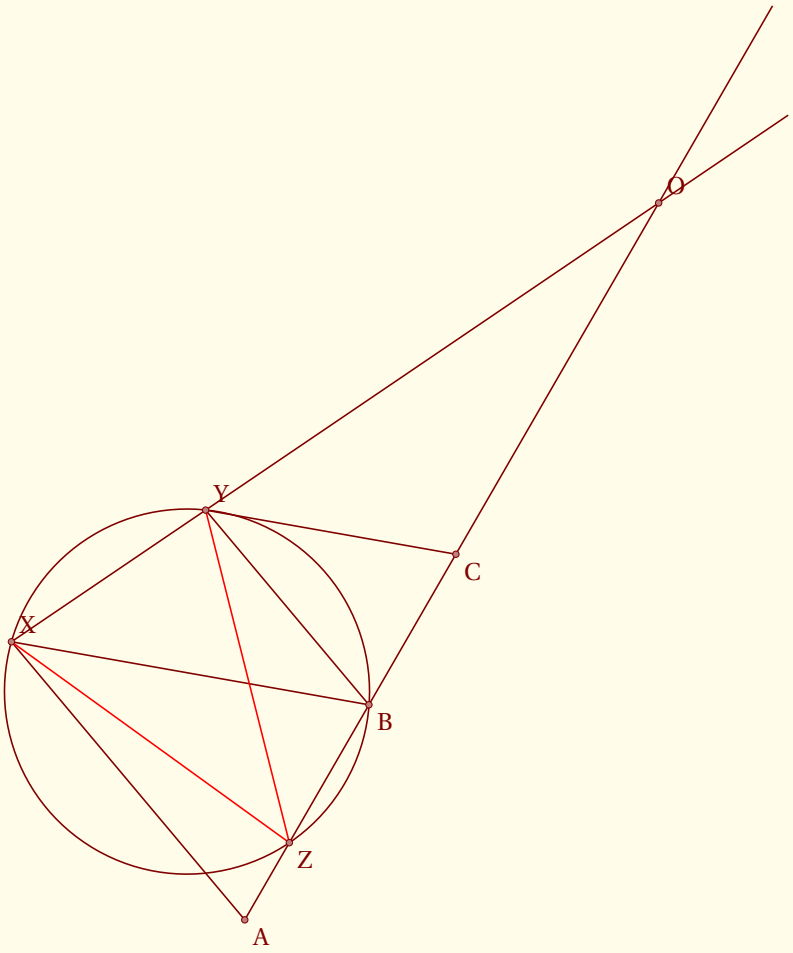
Deux énoncés ont été proposés, l'un par la revue *Tangente*, et l'autre par le journal *Le Monde*.

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 ?

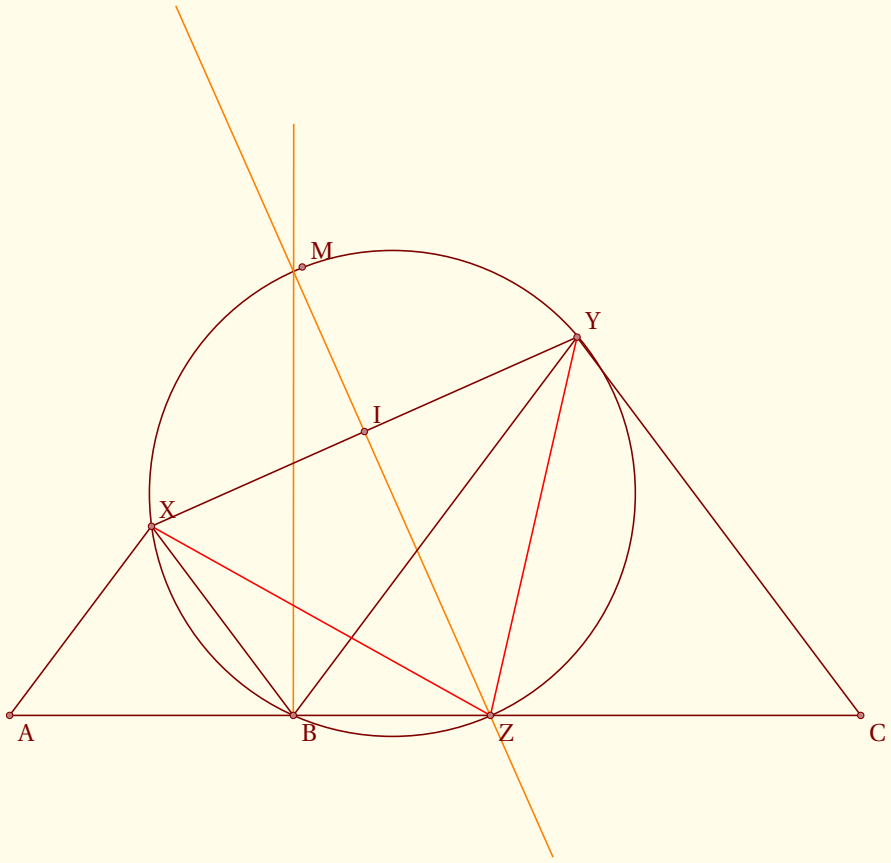
Les constructions et leurs codes associés sont sur les deux pages suivantes, mais vous pouvez chercher avant de regarder. La programmation respecte (il me semble ...), mon raisonnement dans les deux cas.

21.1.1 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}
  \tkzDrawCircle[circum](X,Y,B)
  \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}
```

21.1.2 version "Le Monde"



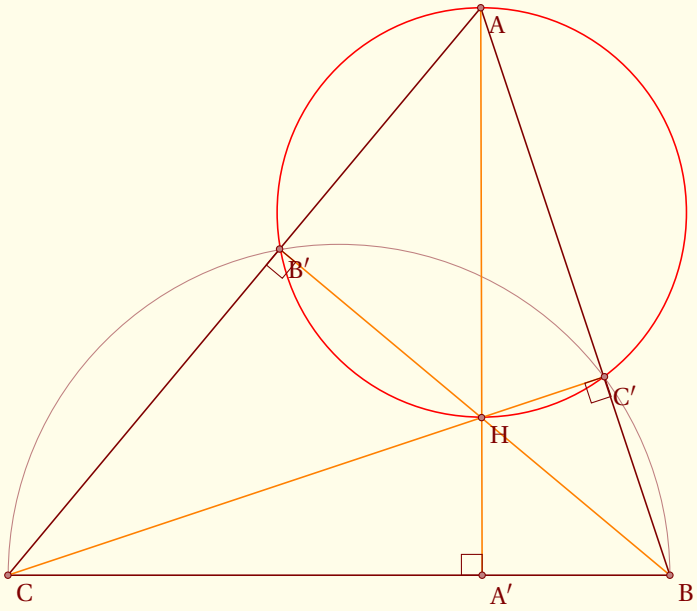

```
\begin{tikzpicture}[scale=1.25]
  \tkzDefPoint(0,0){A}
  \tkzDefPoint(3,0){B}
  \tkzDefPoint(9,0){C}
  \tkzDefPoint(1.5,2){X}
  \tkzDefPoint(6,4){Y}
   \tkzDefCircle[circum](X,Y,B) \tkzGetPoint{O}
  \tkzDefMidPoint(X,Y)               \tkzGetPoint{I}
  \tkzDefPointWith[orthogonal](I,Y)  \tkzGetPoint{i}
  \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}
```

21.2 Hauteurs d'un triangle

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

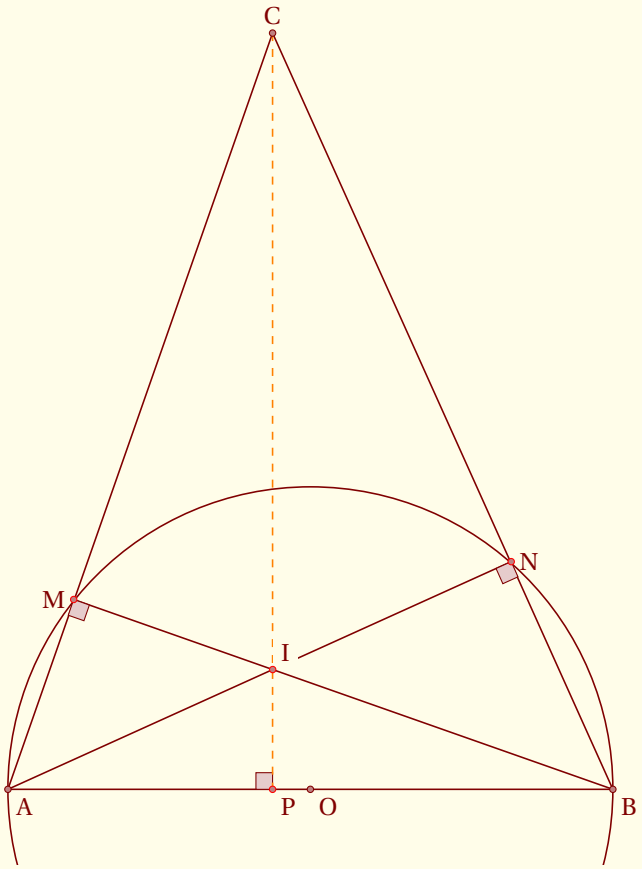
http://debart.pagesperso-orange.fr/geoplan/geometrie_triangle.html

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




```
\begin{tikzpicture}[scale=1.25]
  \tkzInit[xmin= 0,xmax=8 ,ymin=0 ,ymax=7 ] \tkzClip[space=.5]
   \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'}
   \tkzDrawCircle[circum,color=red](A,B',C')
   \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}
```

21.3 Hauteurs - autre construction




```
\begin{tikzpicture}
  \tkzClip[space=1]
  \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}
```

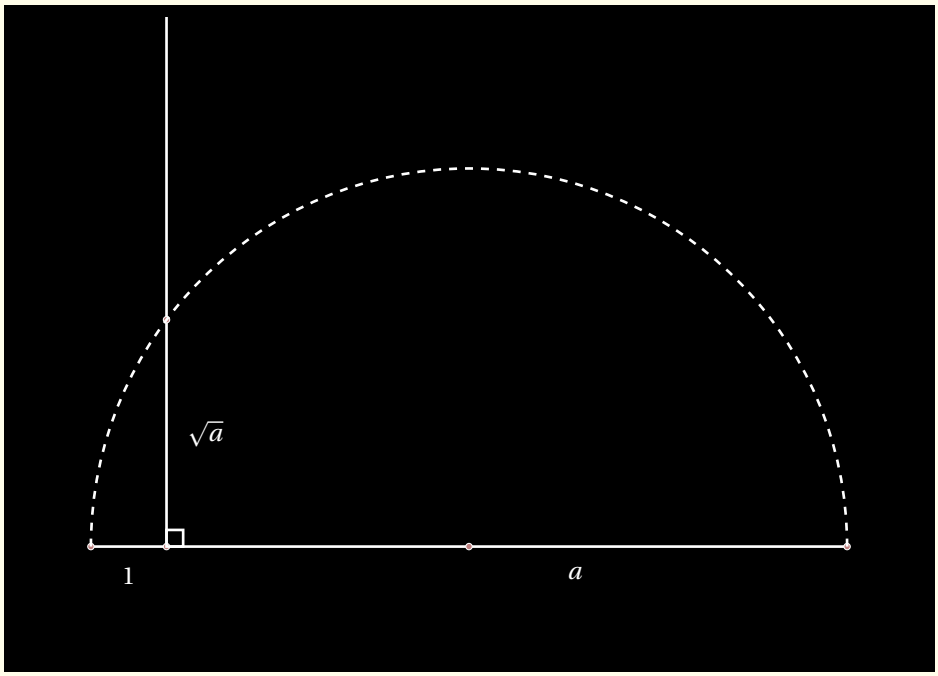
SECTION 22

Gallery : Some examples

Some examples with explanations in english.

22.1 White on Black

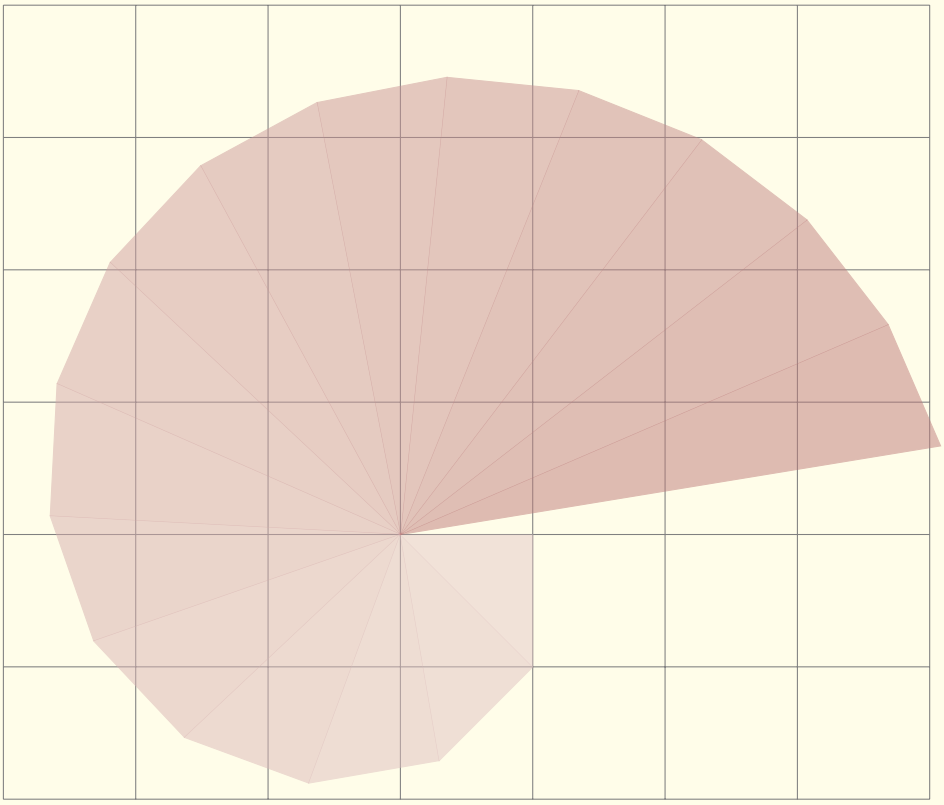
This example shows how to get a segment with a length equal at $\sqrt{a}$ from a segment of length a, only with a rule and a compass.




```
\begin{tikzpicture}[show background rectangle]
   \tkzInit[ymin=-1.5,ymax=7,xmin=-1,xmax=+11]
   \tkzClip
   \tkzDefPoint(0,0){O}
   \tkzDefPoint(1,0){I}
   \tkzDefPoint(10,0){A}
   \tkzDefPointWith[orthogonal](I,A) \tkzGetPoint{H}
   \tkzDefMidPoint(O,A) \tkzGetPoint{M}
   \tkzInterLC(I,H)(M,A)\tkzGetPoints{C}{B}
   \tkzDrawSegments[color=white,line width=1pt](I,H O,A)
   \tkzDrawPoints[color=white](O,I,A,B,M)
   \tkzMarkRightAngle[color=white,line width=1pt](A,I,B)
   \tkzDrawArc[color=white,line width=1pt,style=dashed](M,A)(O)
  \tkzLabelSegment[white,right=1ex,pos=.5](I,B){$\sqrt{a}$}
  \tkzLabelSegment[white,below=1ex,pos=.5](O,I){$1$}
  \tkzLabelSegment[pos=.6,white,below=1ex](I,A){$a$}
\end{tikzpicture}
```

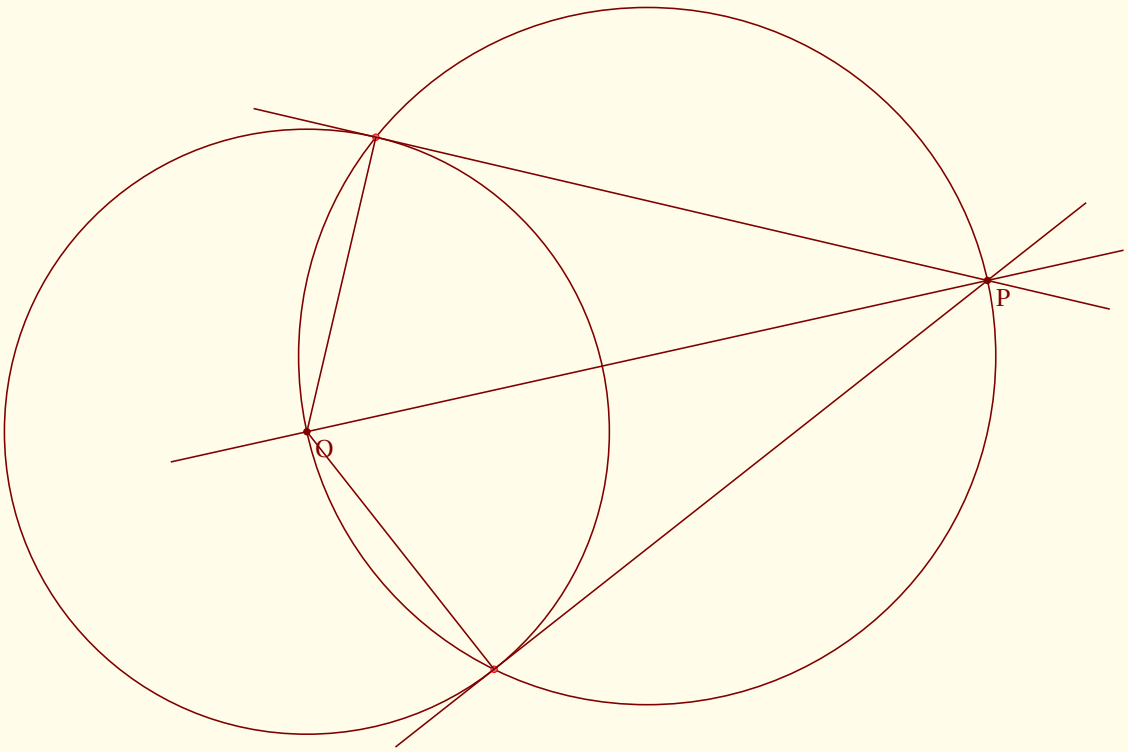
22.2 Square root of the integers

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



```
\begin{tikzpicture}[scale=1.75]
   \tkzInit[xmin=-3,xmax=4,ymin=-2,ymax=4]
   \tkzGrid
   \tkzDefPoint(0,0){O}
   \tkzDefPoint(1,0){a0}
   \newcounter{tkzcounter}
   \setcounter{tkzcounter}{0}
   \newcounter{density}
   \setcounter{density}{20}
   \foreach \i in {0,...,15}{%
      \pgfmathsetcounter{density}{\thedensity+2}
      \setcounter{density}{\thedensity}
      \stepcounter{tkzcounter}
      \tkzDefPointWith[orthogonal normed](a\i,O)
      \tkzGetPoint{a\thetkzcounter}
      \tkzDrawPolySeg[color=Maroon!\thedensity,%
         fill=Maroon!\thedensity,opacity=.5](a\i,a\thetkzcounter,O)}
 \end{tikzpicture}
```

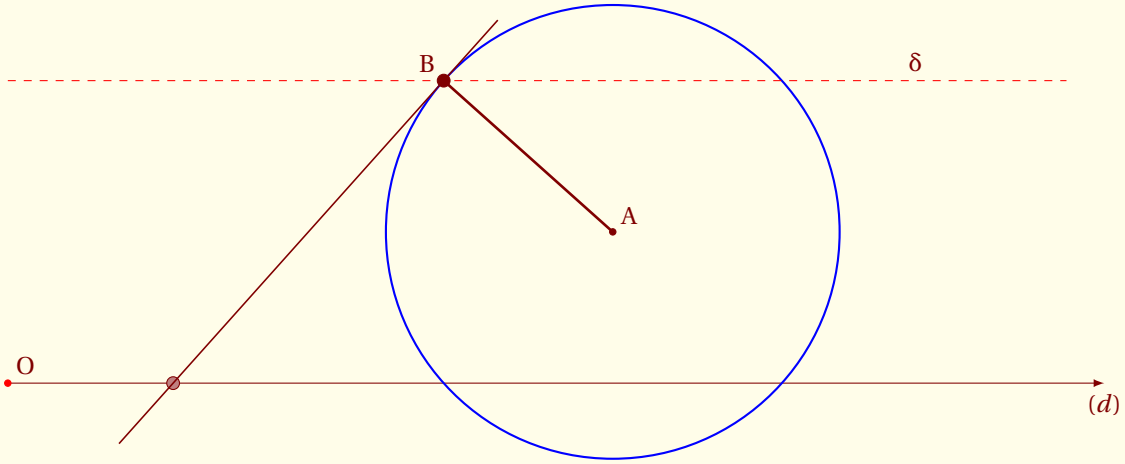
22.3 How to construct the tangent lines from a point to a circle with a rule and a compass.




```
  \begin{tikzpicture}
    \tkzPoint(0,0){O}
    \tkzPoint(9,2){P}
    \tkzDefMidPoint(O,P) \tkzGetPoint{I}
    \tkzDrawCircle[R](O,4cm)
    \tkzDrawCircle[diameter](O,P)
    \tkzCalcLength(I,P)  \tkzGetLength{dIP}
    \tkzInterCC[R](O,4cm)(I,\dIP pt)\tkzGetPoints{Q1}{Q2}
    \tkzDrawPoint[color=red](Q1)
    \tkzDrawPoint[color=red](Q2)
    \tkzDrawLine(P,Q1)
    \tkzDrawLine(P,Q2)
    \tkzDrawSegments(O,Q1 O,Q2)
    \tkzDrawLine(P,O)
\end{tikzpicture}
```

22.4 Circle and tangent

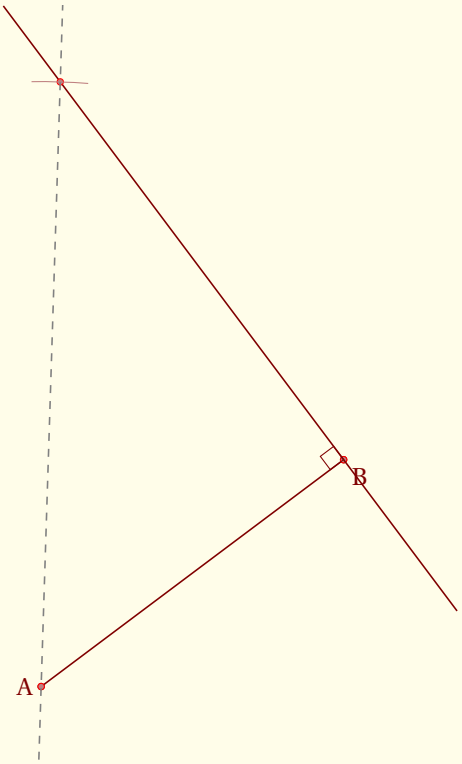
We have a point A (8,2), a circle with center A and radius=3cm and a line δ $y = 4$. The line intercepts the circle at B. We want to draw the tangent at the circle in B.




```
\begin{tikzpicture}
  \tkzInit[xmax=14,ymin=-2,ymax=6]
  \tkzDrawX[noticks,label=$(d)$]
  \tkzPoint[pos=above right](8,2){A};
  \tkzPoint[color=red,pos=above right](0,0){O};
  \tkzDrawCircle[R,color=blue,line width=.8pt](A,3 cm)
  \tkzHLine[color=red,style=dashed]{4}
  \tkzText[above](12,4){$\delta$}
  \FPeval\alphaR{arcsin(2/3)}% on a les bonnes valeurs
  \FPeval\xB{8-3*cos(\alphaR)}
  \tkzPoint[pos=above left](\xB,4){B};
  \tkzDrawSegment[line width=1pt](A,B)
  \tkzDefLine[orthogonal=through B](A,B) \tkzGetPoint{b}
  \tkzDefPoint(1,0){i}
  \tkzInterLL(B,b)(O,i) \tkzGetPoint{B'}
  \tkzDrawPoint(B')
  \tkzDrawLine(B,B')
 \end{tikzpicture}
```

22.5 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.

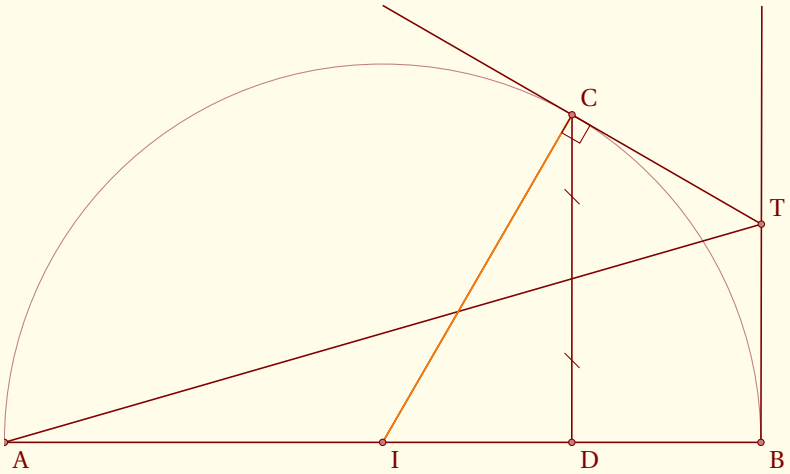



```
\begin{tikzpicture}
  \tkzInit
  \tkzClip
  \tkzPoint[pos=left](2,1){A}
  \tkzPoint(6,4){B}
  \tkzDrawSegment(A,B)
  \tkzDrawPoint[color=red](A)
  \tkzDrawPoint[color=red](B)
  \tkzDefPointWith[orthogonal,K=-1](B,A)
  \tkzDrawLine[add = .5 and .5](B,tkzPointResult)
  \tkzInterLC[R](B,tkzPointResult)(A,8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

22.6 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.

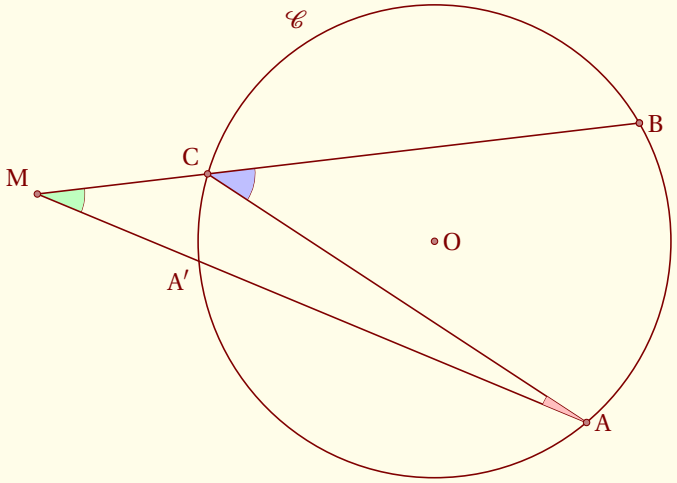
Prove that the line (AT) bisects (CD)




```
\begin{tikzpicture}[scale=1.25]
   \tkzInit[ymin=-1,ymax=7]
   \tkzClip
   \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}
```

22.7 Example from 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**.

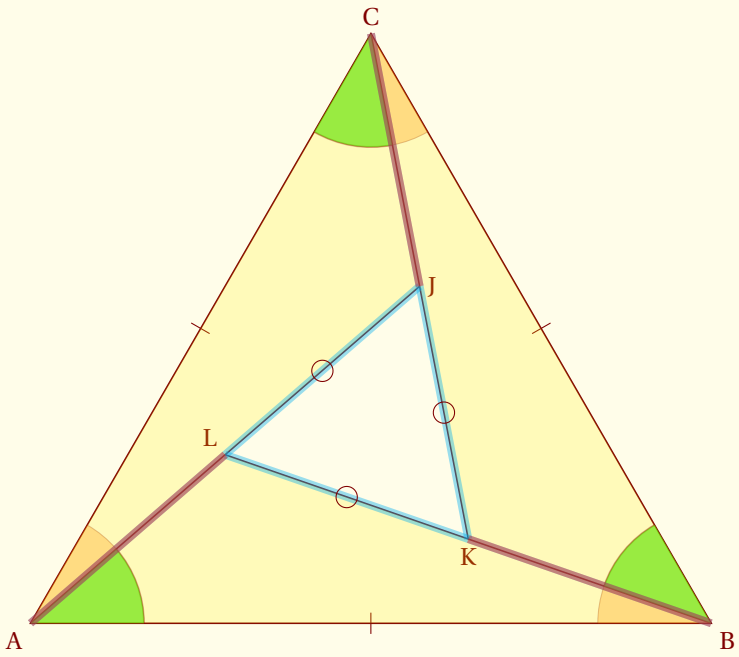



```
\begin{tikzpicture}[scale=1.25]
  \tkzInit[xmin=-5.2,xmax=3.2,ymin=-3.2,ymax=3.3]
  \tkzClip
  \tkzDefPoint(0,0){O}
  \tkzDefPoint(2.5,0){N}
  \tkzDefPoint(-4.2,0.5){M}
  \tkzDefPointBy[rotation=center O angle 30](N)
  \tkzGetPoint{B}
  \tkzDefPointBy[rotation=center O angle -50](N)
  \tkzGetPoint{A}
  \tkzInterLC(M,B)(O,N) \tkzGetFirstPoint{C}
  \tkzInterLC(M,A)(O,N) \tkzGetSecondPoint{A'}
  \tkzMarkAngle[fill=blue!25,mkpos=.2, size=0.5](A,C,B)
  \tkzMarkAngle[fill=green!25,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[fill=red!25,mkpos=.2, size=0.5cm](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}
```

22.8 Example 1 from John Kitzmiller

This figure is the last of beamer document. You can find the document on my site

Prove △LKJ is equilateral

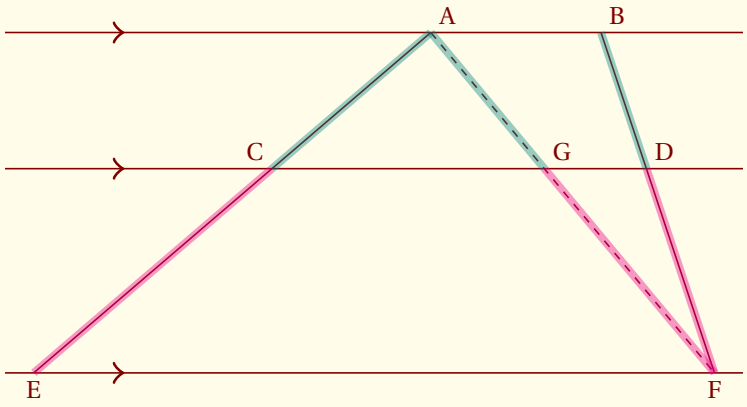



```
\begin{tikzpicture}[scale=1.5]
  \tkzDefPoint[label=below left:A](0,0){A}
  \tkzDefPoint[label=below right:B](6,0){B}
  \tkzDefTriangle[equilateral](A,B) \tkzGetPoint{C}
  \tkzMarkSegments[mark=|](A,B A,C B,C)
  \tkzDefBarycentricPoint(A=1,B=2) \tkzGetPoint{C'}
  \tkzDefBarycentricPoint(A=2,C=1) \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}
```

22.9 Example 2 from John Kitzmiller

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

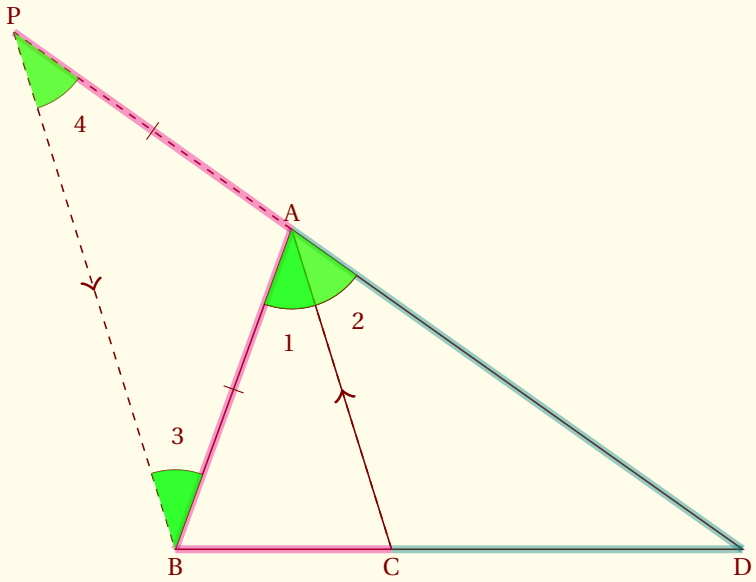
Another interesting example from John, you can see how to use some extra options like **decoration** and **postaction** from **TikZ** with **tkz-euclide**.




```
\begin{tikzpicture}[scale=1.5,decoration={markings,
  mark=at position 3cm with {\arrow[scale=2]{>}};}]
  \tkzInit[xmin=-0.25,xmax=6.25, ymin=-0.5,ymax=4]
  \tkzClip
  \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}
```

22.10 Example 3 from John Kitzmiller

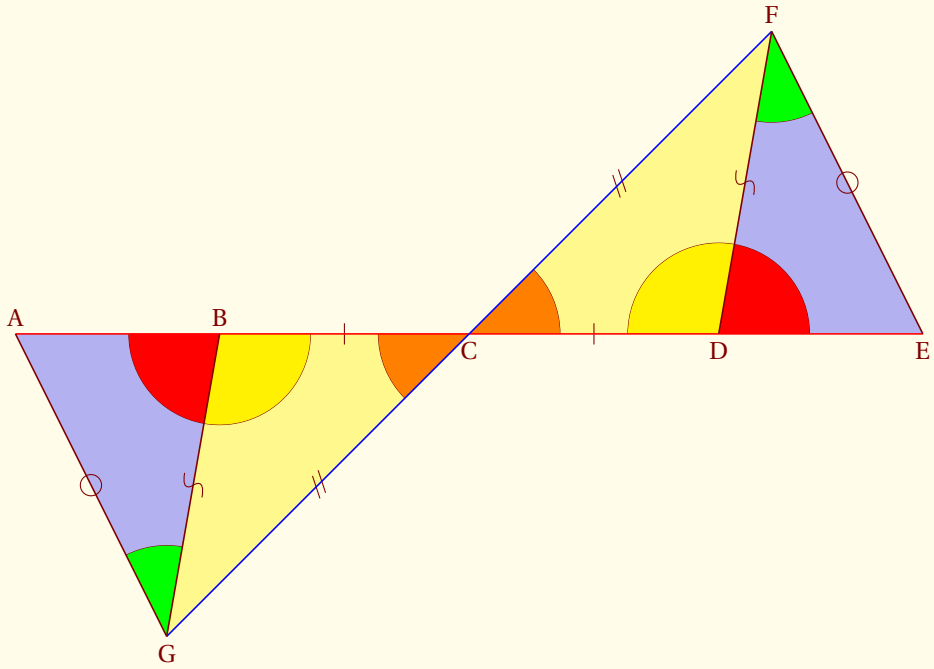
Prove $\frac{BC}{CD} = \frac{AB}{AD}$ (Angle Bisector)




```
\begin{tikzpicture}[scale=1.5]
  \tkzInit[xmin=-4,xmax=5,ymax=4.5]   \tkzClip[space=.5]
  \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}
```

22.11 Example 4 from John Kitzmiller

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




```
\begin{tikzpicture}[scale=2]
  \tkzInit[xmax=5, ymax=5]
  \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}
```

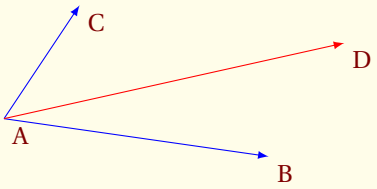
SECTION 23

FAQ

23.1 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
- 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 **fp.sty**. J'ai choisi souvent **fp.sty** 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 erreurs. Ces erreurs 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 seul **\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**
- Somme de deux vecteurs
  Comment obtenir le point D tel que $\overrightarrow{AD} = \overrightarrow{AB} + \overrightarrow{AC}$?




```
  \begin{tikzpicture}[scale=.5]
  \tkzDefPoint(1,1){A}
  \tkzDefPoint(8,0){B}
  \tkzDefPoint(3,4){C}
  \tkzDefVector[colinear= at C](A,B){D}
  \tkzDrawVectors[color=blue](A,B A,C)
  \tkzDrawVector[color=red](A,D)
  \tkzLabelPoints(A,B,C,D)
\end{tikzpicture}
```

Index

U