

# TikZ & PGF Introduction

## Les commandes de base

Bertrand Masson

Les fiches de Bébert

15 octobre 2009

Introduction

Le système de  
coordonnées

Les coordonnées  
relatives

Les coordonnées  
relatives

Les chemins : path

Les opérations

Les actions

La couleur

Position des options

Les actions (suite)

conclusion

- 1 Introduction
- 2 Le système de coordonnées
- 3 Les opérations
- 4 Les actions
- 5 La couleur
- 6 Position des options
- 7 Les actions (suite)
- 8 conclusion

## Le système TikZ & PGF

**PGF** est une extension servant à la réalisation de graphique, comme  $\text{T}_{\text{E}}\text{X}$  l'est à la réalisation de mise en page de texte.

**TikZ** est une interface permettant de faciliter l'utilisation de **PGF**, comme  $\text{\LaTeX}$  pour  $\text{T}_{\text{E}}\text{X}$ .

Tu utilises **TikZ & PGF** dans un source  $\text{\LaTeX}$  en chargeant le package **TikZ** (`\usepackage{tikz}`).

Le package **TikZ** charge automatiquement le package **xcolor** donc inutile de le charger.

## Le système de coordonnées

Pour faire ses dessins TiKZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (**angle** : **rayon**), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le **centimètre** ; l'unité d'angle est le **degré**. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

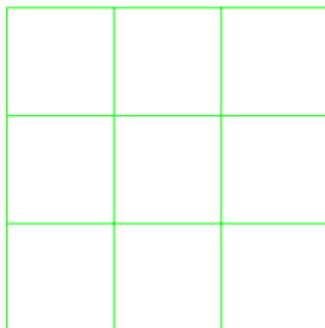
**Les coordonnées cartésiennes :**

## Le système de coordonnées

Pour faire ses dessins TikZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (angle : rayon), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le centimètre ; l'unité d'angle est le degré. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

**Les coordonnées cartésiennes :**

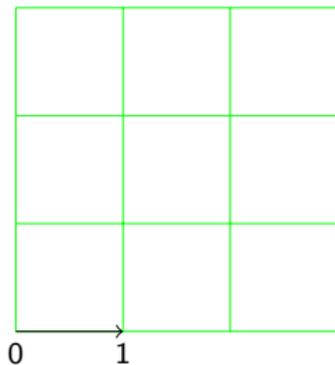
Les  $x$  augmentent vers la droite



## Le système de coordonnées

Pour faire ses dessins TiKZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (angle : rayon), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le centimètre ; l'unité d'angle est le degré. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

**Les coordonnées cartésiennes :**  
Les  $x$  augmentent vers la droite

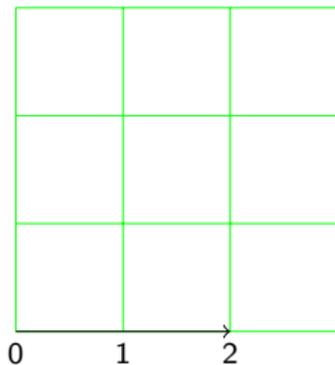


## Le système de coordonnées

Pour faire ses dessins TiKZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (angle : rayon), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le centimètre ; l'unité d'angle est le degré. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

**Les coordonnées cartésiennes :**

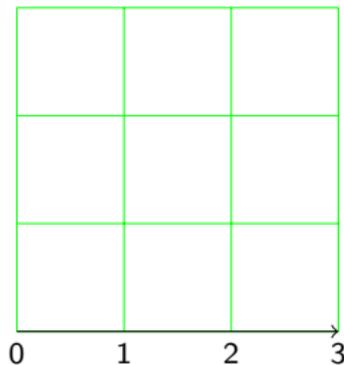
Les  $x$  augmentent vers la droite



## Le système de coordonnées

Pour faire ses dessins TikZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (angle : rayon), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le centimètre ; l'unité d'angle est le degré. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

**Les coordonnées cartésiennes :**  
Les  $x$  augmentent vers la droite

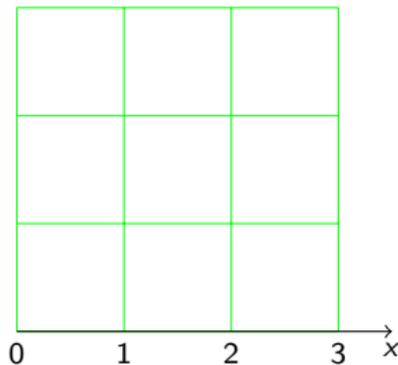


## Le système de coordonnées

Pour faire ses dessins TiKZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (angle : rayon), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le centimètre ; l'unité d'angle est le degré. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

**Les coordonnées cartésiennes :**

Les  $x$  augmentent vers la droite

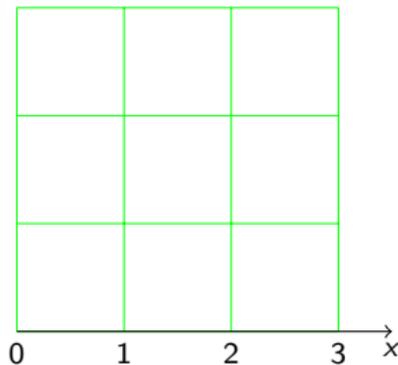


## Le système de coordonnées

Pour faire ses dessins TiKZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (angle : rayon), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le centimètre ; l'unité d'angle est le degré. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

**Les coordonnées cartésiennes :**

Les  $x$  augmentent vers la droite

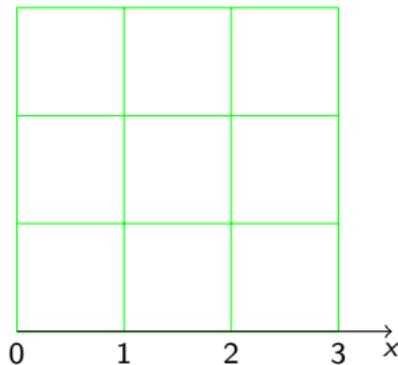


## Le système de coordonnées

Pour faire ses dessins TiKZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (angle : rayon), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le centimètre ; l'unité d'angle est le degré. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

### Les coordonnées cartésiennes :

Les  $x$  augmentent vers la droite et les  $y$  vers le haut,

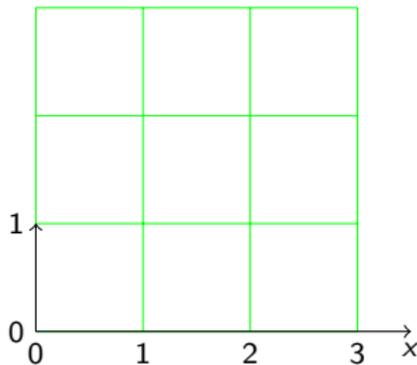


## Le système de coordonnées

Pour faire ses dessins TiKZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (angle : rayon), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le centimètre ; l'unité d'angle est le degré. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

### Les coordonnées cartésiennes :

Les  $x$  augmentent vers la droite et les  $y$  vers le haut,

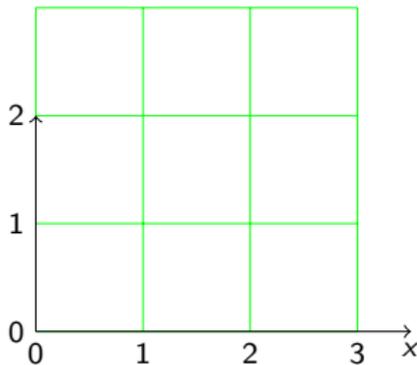


## Le système de coordonnées

Pour faire ses dessins TiKZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (angle : rayon), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le centimètre ; l'unité d'angle est le degré. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

### Les coordonnées cartésiennes :

Les  $x$  augmentent vers la droite et les  $y$  vers le haut,

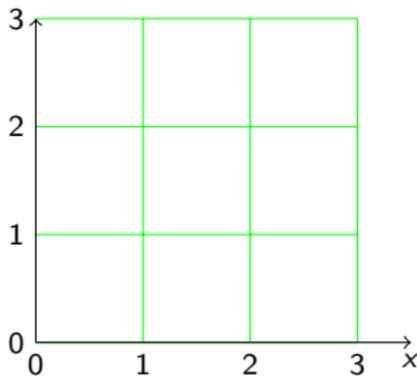


## Le système de coordonnées

Pour faire ses dessins TiKZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (angle : rayon), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le centimètre ; l'unité d'angle est le degré. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

### Les coordonnées cartésiennes :

Les  $x$  augmentent vers la droite et les  $y$  vers le haut,

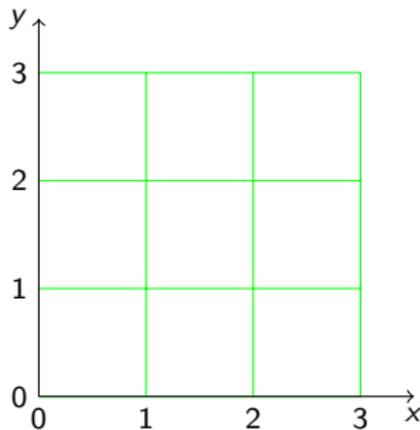


## Le système de coordonnées

Pour faire ses dessins TiKZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (angle : rayon), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le centimètre ; l'unité d'angle est le degré. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

### Les coordonnées cartésiennes :

Les  $x$  augmentent vers la droite et les  $y$  vers le haut,

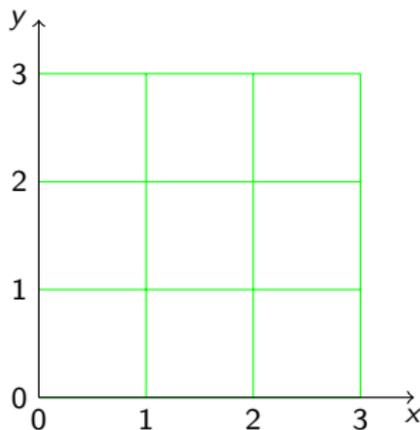


## Le système de coordonnées

Pour faire ses dessins TiKZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (angle : rayon), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le centimètre ; l'unité d'angle est le degré. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

### Les coordonnées cartésiennes :

Les  $x$  augmentent vers la droite et les  $y$  vers le haut,

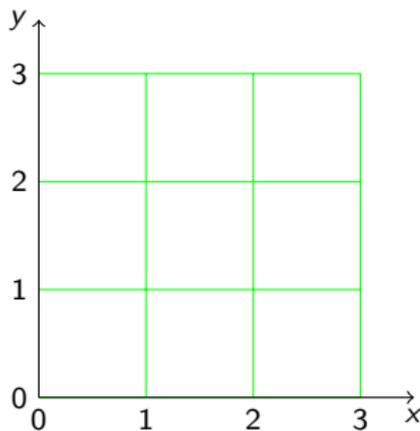


## Le système de coordonnées

Pour faire ses dessins TiKZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (angle : rayon), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le centimètre ; l'unité d'angle est le degré. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

### Les coordonnées cartésiennes :

Les  $x$  augmentent vers la droite et les  $y$  vers le haut, l'origine est donc en bas à gauche.

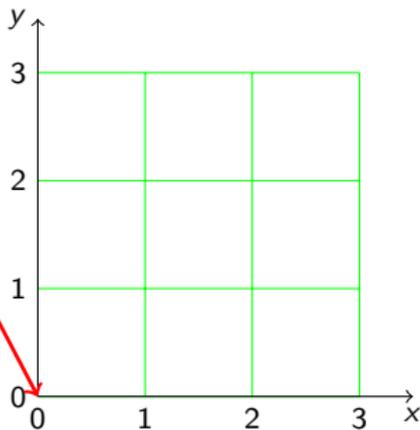


## Le système de coordonnées

Pour faire ses dessins TiKZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (angle : rayon), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le centimètre ; l'unité d'angle est le degré. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

### Les coordonnées cartésiennes :

Les  $x$  augmentent vers la droite et les  $y$  vers le haut, l'origine est donc en bas à gauche.



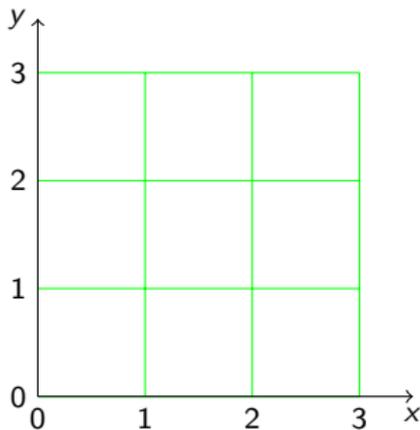
## Le système de coordonnées

Pour faire ses dessins TiKZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (angle : rayon), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le centimètre ; l'unité d'angle est le degré. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

### Les coordonnées cartésiennes :

Les  $x$  augmentent vers la droite et les  $y$  vers le haut, l'origine est donc en bas à gauche.

Les coordonnées s'écrivent toujours entre parenthèses  $()$ . Par exemple voici un point rouge à 2 cm en  $x$  et 1 cm en  $y$   $(2,1)$ ,



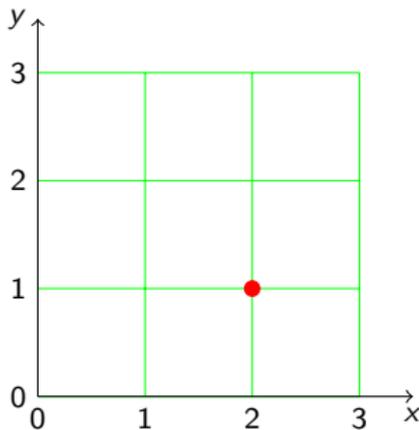
## Le système de coordonnées

Pour faire ses dessins TiKZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (angle : rayon), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le centimètre ; l'unité d'angle est le degré. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

### Les coordonnées cartésiennes :

Les  $x$  augmentent vers la droite et les  $y$  vers le haut, l'origine est donc en bas à gauche.

Les coordonnées s'écrivent toujours entre parenthèses  $()$ . Par exemple voici un point rouge à 2 cm en  $x$  et 1 cm en  $y$   $(2,1)$ ,



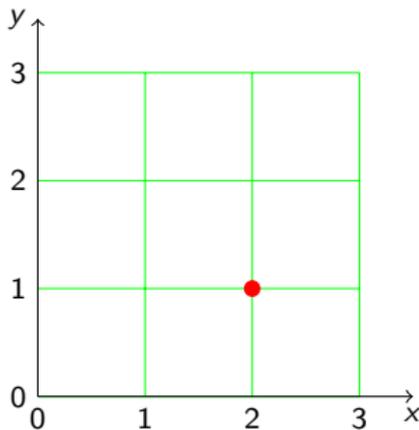
## Le système de coordonnées

Pour faire ses dessins TiKZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (**angle** : **rayon**), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le **centimètre** ; l'unité d'angle est le **degré**. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

### Les coordonnées cartésiennes :

Les  $x$  augmentent vers la droite et les  $y$  vers le haut, l'origine est donc en bas à gauche.

Les coordonnées s'écrivent toujours entre parenthèses  $()$ . Par exemple voici un point rouge à 2 cm en  $x$  et 1 cm en  $y$  ( $2,1$ ), un point bleu à  $(1.3,2.75)$ .



## Le système de coordonnées

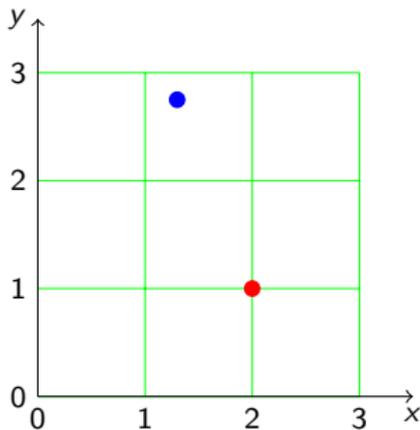
Pour faire ses dessins TiKZ utilise plusieurs systèmes de coordonnées : cartésiennes  $(x, y)$ , polaires (**angle** : **rayon**), des coordonnées en XYZ et des coordonnées barycentriques. N'utilisant que les coordonnées cartésiennes, je ne décrirais que ses dernières. L'unité de longueur par défaut est le **centimètre** ; l'unité d'angle est le **degré**. Si tu ne précises pas les unités ce sont celles par défauts qui sont utilisées.

### Les coordonnées cartésiennes :

Les  $x$  augmentent vers la droite et les  $y$  vers le haut, l'origine est donc en bas à gauche.

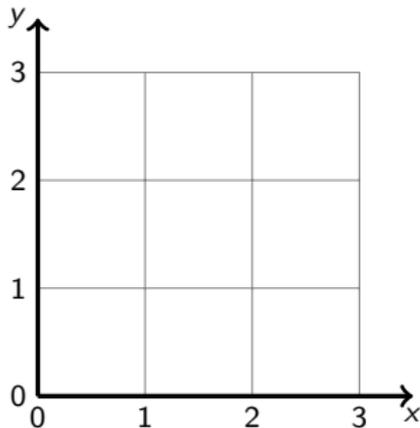
Les coordonnées s'écrivent toujours entre parenthèses  $()$ . Par exemple voici un point rouge à 2 cm en  $x$  et 1 cm en  $y$   $(2,1)$ , un point bleu à  $(1.3,2.75)$ .

**Attention on note à l'anglo-saxonne donc le marqueur de décimale est le point et le séparateur de coordonnées la virgule.**



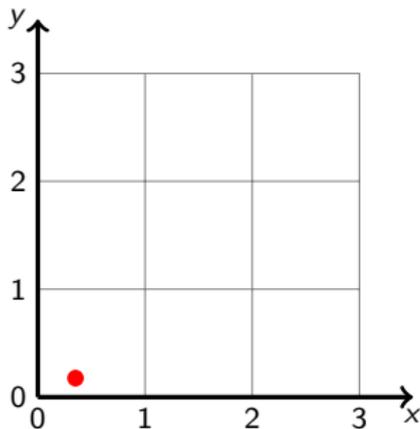
## Les unités

Toutes les unités comprise par  $\text{\LaTeX}$  peuvent être utilisées (voir la fiche «  $\text{\LaTeX}$  les unités & les longueurs »). Dans le cas où tu n'utilises pas le centimètre, il faut préciser les unités, par exemple un point rouge à  $(10\text{pt},5\text{pt})$ .



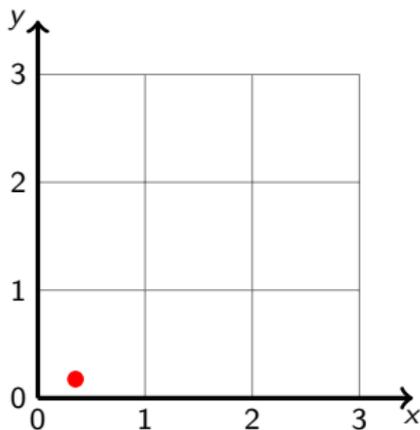
## Les unités

Toutes les unités comprise par  $\text{\LaTeX}$  peuvent être utilisées (voir la fiche «  $\text{\LaTeX}$  les unités & les longueurs »). Dans le cas où tu n'utilises pas le centimètre, il faut préciser les unités, par exemple un point rouge à  $(10\text{pt},5\text{pt})$ .



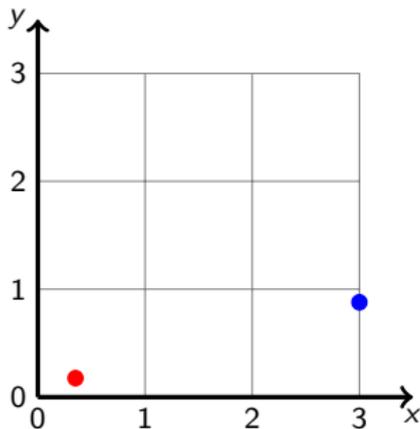
## Les unités

Toutes les unités comprise par  $\text{\LaTeX}$  peuvent être utilisées (voir la fiche «  $\text{\LaTeX}$  les unités & les longueurs »). Dans le cas où tu n'utilises pas le centimètre, il faut préciser les unités, par exemple un point rouge à  $(10\text{pt},5\text{pt})$ . Tu peux mélanger les unités, un point bleu à  $(30\text{mm},25\text{pt})$ .



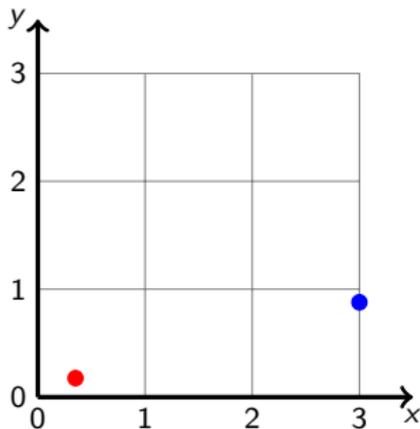
## Les unités

Toutes les unités comprise par  $\LaTeX$  peuvent être utilisées (voir la fiche «  $\LaTeX$  les unités & les longueurs »). Dans le cas où tu n'utilises pas le centimètre, il faut préciser les unités, par exemple un point rouge à  $(10\text{pt},5\text{pt})$ . Tu peux mélanger les unités, un point bleu à  $(30\text{mm},25\text{pt})$ .



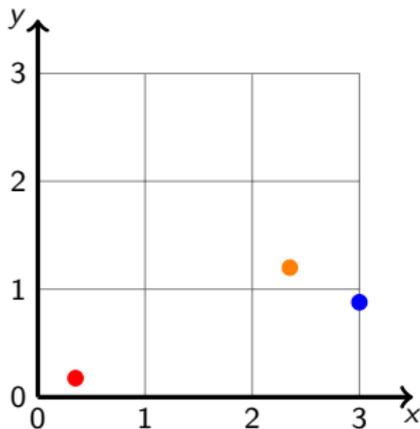
## Les unités

Toutes les unités comprise par  $\LaTeX$  peuvent être utilisées (voir la fiche «  $\LaTeX$  les unités & les longueurs »). Dans le cas où tu n'utilises pas le centimètre, il faut préciser les unités, par exemple un point rouge à  $(10\text{pt},5\text{pt})$ . Tu peux mélanger les unités, un point bleu à  $(30\text{mm},25\text{pt})$ . Tu peux mettre des opérateurs, un point orange à  $(2\text{cm}+10\text{pt},0.6\text{cm}*2)$



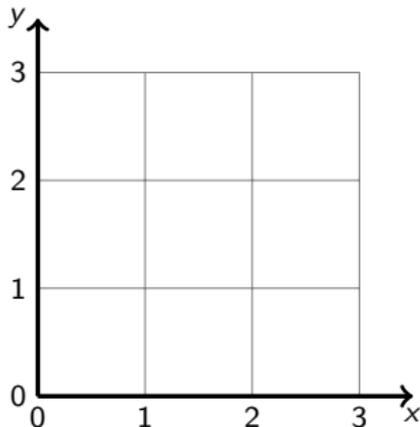
## Les unités

Toutes les unités comprise par  $\LaTeX$  peuvent être utilisées (voir la fiche «  $\LaTeX$  les unités & les longueurs »). Dans le cas où tu n'utilises pas le centimètre, il faut préciser les unités, par exemple un point rouge à  $(10\text{pt},5\text{pt})$ . Tu peux mélanger les unités, un point bleu à  $(30\text{mm},25\text{pt})$ . Tu peux mettre des opérateurs, un point orange à  $(2\text{cm}+10\text{pt},0.6\text{cm}*2)$



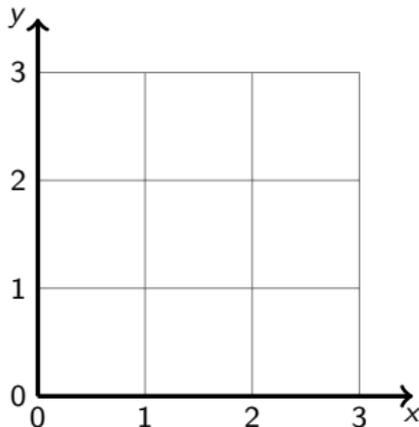
## Les coordonnées relatives

L'opérateur `++` permet de définir des coordonnées par rapport à la coordonnée précédente. Par exemple  $(1,1) ++(1,0) ++(-2,-1)$  signifie,



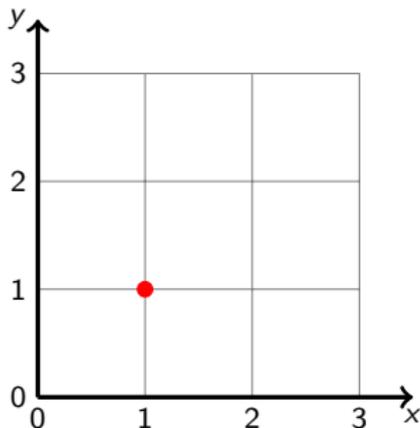
## Les coordonnées relatives

L'opérateur `++` permet de définir des coordonnées par rapport à la coordonnée précédente. Par exemple  $(1,1) ++(1,0) ++(-2,-1)$  signifie, une première coordonnée à  $(1,1)$  symbolisée par le point rouge,



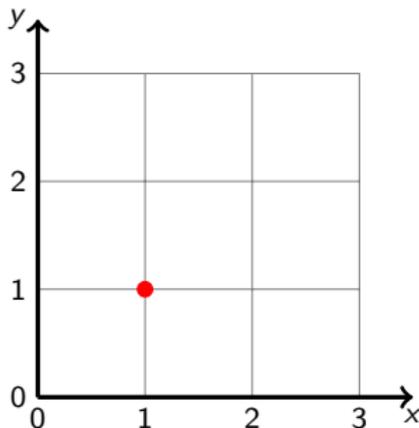
## Les coordonnées relatives

L'opérateur `++` permet de définir des coordonnées par rapport à la coordonnée précédente. Par exemple  $(1,1) ++(1,0) ++(-2,-1)$  signifie, une première coordonnée à  $(1,1)$  symbolisée par le point rouge,



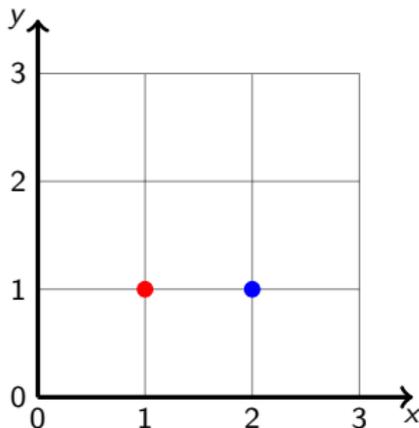
## Les coordonnées relatives

L'opérateur `++` permet de définir des coordonnées par rapport à la coordonnée précédente. Par exemple  $(1,1) ++(1,0) ++(-2,-1)$  signifie, une première coordonnée à  $(1,1)$  symbolisée par le point rouge, la 2<sup>me</sup>  $++(1,0)$  se trouve à 1 cm à gauche de la précédente soit à  $(2,1)$  (point bleu),



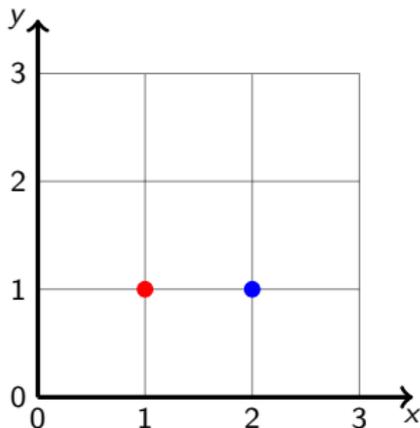
## Les coordonnées relatives

L'opérateur `++` permet de définir des coordonnées par rapport à la coordonnée précédente. Par exemple  $(1,1) ++(1,0) ++(-2,-1)$  signifie, une première coordonnée à  $(1,1)$  symbolisée par le point rouge, la 2<sup>me</sup>  $++(1,0)$  se trouve à 1 cm à gauche de la précédente soit à  $(2,1)$  (point bleu),



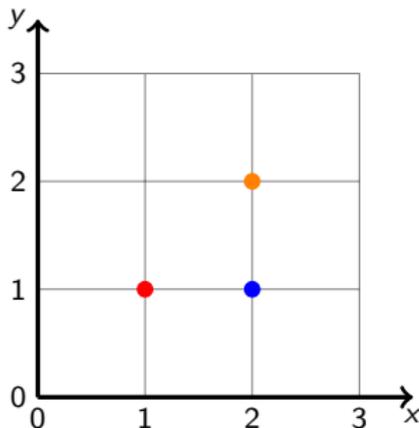
## Les coordonnées relatives

L'opérateur `++` permet de définir des coordonnées par rapport à la coordonnée précédente. Par exemple  $(1,1) ++(1,0) ++(-2,-1)$  signifie, une première coordonnée à  $(1,1)$  symbolisée par le point rouge, la 2<sup>me</sup>  $++(1,0)$  se trouve à 1 cm à gauche de la précédente soit à  $(2,1)$  (point bleu), la suivante à  $++(0,1)$  donc à 1 cm au-dessus de la précédente (donc du point bleu) soit à  $(2,2)$  (point orange)



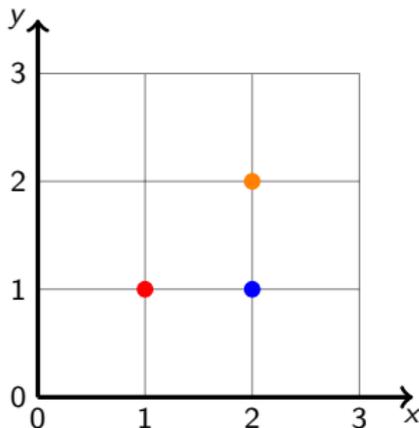
## Les coordonnées relatives

L'opérateur `++` permet de définir des coordonnées par rapport à la coordonnée précédente. Par exemple  $(1,1) ++(1,0) ++(-2,-1)$  signifie, une première coordonnée à  $(1,1)$  symbolisée par le point rouge, la 2<sup>me</sup>  $++(1,0)$  se trouve à 1 cm à gauche de la précédente soit à  $(2,1)$  (point bleu), la suivante à  $++(0,1)$  donc à 1 cm au-dessus de la précédente (donc du point bleu) soit à  $(2,2)$  (point orange)



## Les coordonnées relatives

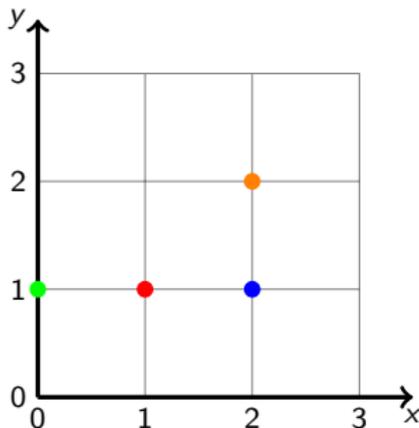
L'opérateur `++` permet de définir des coordonnées par rapport à la coordonnée précédente. Par exemple  $(1,1) ++(1,0) ++(-2,-1)$  signifie, une première coordonnée à  $(1,1)$  symbolisée par le point rouge, la 2<sup>me</sup>  $++(1,0)$  se trouve à 1 cm à gauche de la précédente soit à  $(2,1)$  (point bleu), la suivante à  $++(0,1)$  donc à 1 cm au-dessus de la précédente (donc du point bleu) soit à  $(2,2)$  (point orange) et enfin on a  $++(-2,-1)$  soit 2 cm vers la droite et 1 cm en bas par rapport au point orange, donc à  $(0,1)$  (point vert).



## Les coordonnées relatives

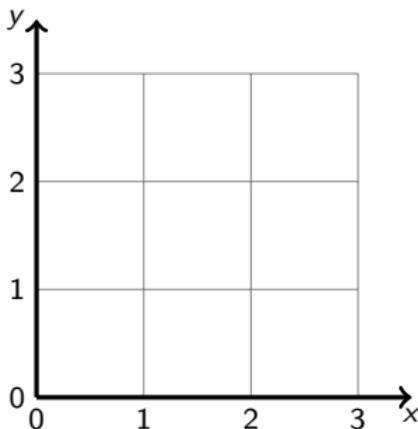
L'opérateur `++` permet de définir des coordonnées par rapport à la coordonnée précédente. Par exemple  $(1,1) ++(1,0) ++(-2,-1)$  signifie, une première coordonnée à  $(1,1)$  symbolisée par le point rouge, la 2<sup>me</sup>  $++(1,0)$  se trouve à 1 cm à gauche de la précédente soit à  $(2,1)$  (point bleu), la suivante à  $++(0,1)$  donc à 1 cm au-dessus de la précédente (donc du point bleu) soit à  $(2,2)$  (point orange) et enfin on a  $++(-2,-1)$  soit 2 cm vers la droite et 1 cm en bas par rapport au point orange, donc à  $(0,1)$  (point vert).

Tu additionnes les x entre-eux et les y entre-eux. Pour le point bleu  $(1,1)++(1,0)$  on a  $1+1=2$  et  $1+0=1$  il est donc à  $(2,1)$ . Pour le point vert  $(2,2)++(-2,-1)$  :  $2+(-2)=0$  et  $2+(-1)=1$  donc à  $(0,1)$ .



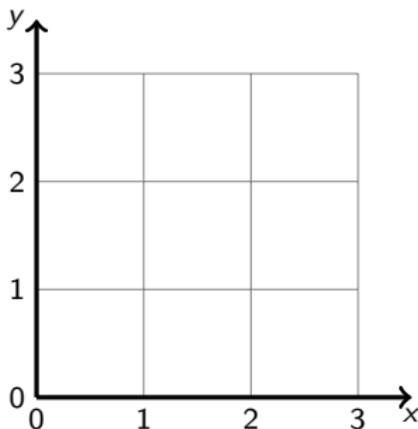
## Les coordonnées relatives

L'opérateur  $+$  se rapporte toujours à la première coordonnée définie. Si on reprend notre exemple précédent  $(1,1) + (1,0) + (-2,-1)$ ,



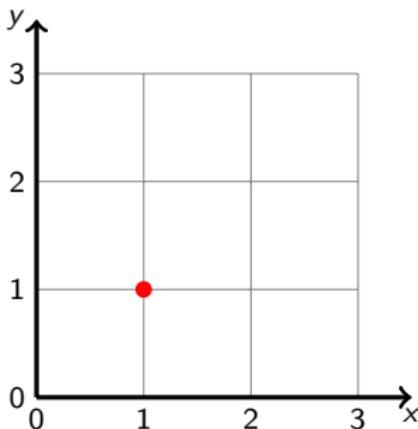
## Les coordonnées relatives

L'opérateur  $+$  se rapporte toujours à la première coordonnée définie. Si on reprend notre exemple précédent  $(1,1) + (1,0) + (-2,-1)$ , on a toujours une première coordonnée à  $(1,1)$  symbolisée par le point rouge,



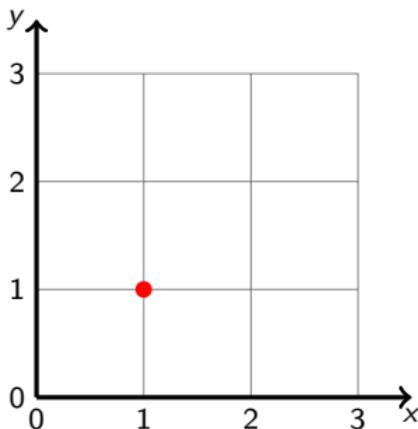
## Les coordonnées relatives

L'opérateur  $+$  se rapporte toujours à la première coordonnée définie. Si on reprend notre exemple précédent  $(1,1) + (1,0) + (-2,-1)$ , on a toujours une première coordonnée à  $(1,1)$  symbolisée par le point rouge,



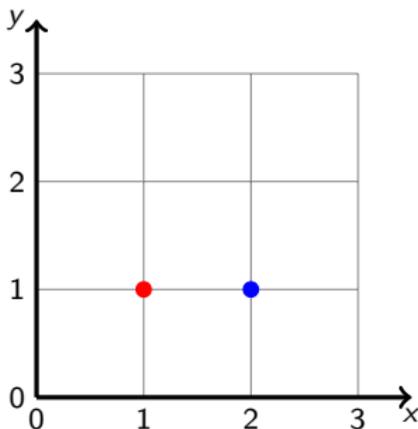
## Les coordonnées relatives

L'opérateur  $+$  se rapporte toujours à la première coordonnée définie. Si on reprend notre exemple précédent  $(1,1) + (1,0) + (-2,-1)$ , on a toujours une première coordonnée à  $(1,1)$  symbolisée par le point rouge, la 2<sup>me</sup>  $+(1,0)$  se trouve à 1 cm à gauche de la première (rouge) soit à  $(2,1)$  (point bleu),



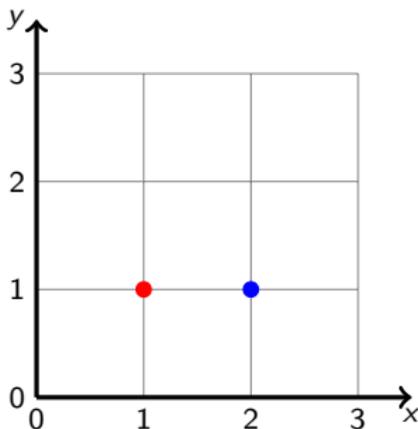
## Les coordonnées relatives

L'opérateur  $+$  se rapporte toujours à la première coordonnée définie. Si on reprend notre exemple précédent  $(1,1) + (1,0) + (-2,-1)$ , on a toujours une première coordonnée à  $(1,1)$  symbolisée par le point rouge, la 2<sup>me</sup>  $+(1,0)$  se trouve à 1 cm à gauche de la première (rouge) soit à  $(2,1)$  (point bleu),



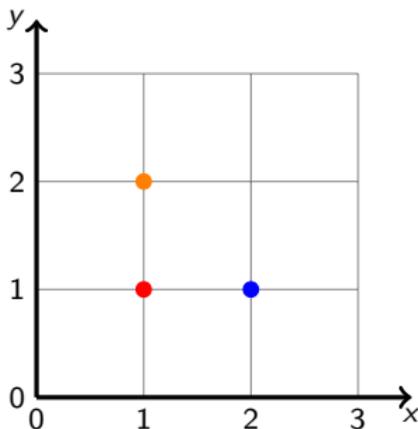
## Les coordonnées relatives

L'opérateur  $+$  se rapporte toujours à la première coordonnée définie. Si on reprend notre exemple précédent  $(1,1) + (1,0) + (-2,-1)$ , on a toujours une première coordonnée à  $(1,1)$  symbolisée par le point rouge, la 2<sup>me</sup>  $+(1,0)$  se trouve à 1 cm à gauche de la première (rouge) soit à  $(2,1)$  (point bleu), la suivante à  $+(0,1)$  donc à 1 cm au-dessus de la rouge soit à  $(1,2)$  (point orange)



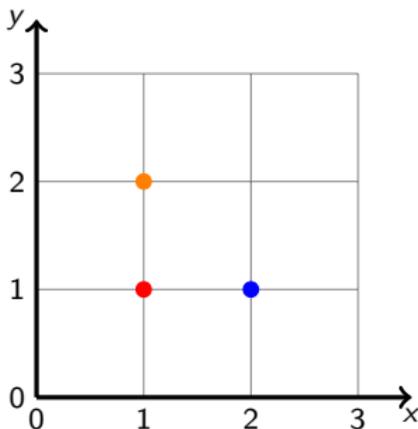
## Les coordonnées relatives

L'opérateur  $+$  se rapporte toujours à la première coordonnée définie. Si on reprend notre exemple précédent  $(1,1) + (1,0) + (-2,-1)$ , on a toujours une première coordonnée à  $(1,1)$  symbolisée par le point rouge, la 2<sup>me</sup>  $+(1,0)$  se trouve à 1 cm à gauche de la première (rouge) soit à  $(2,1)$  (point bleu), la suivante à  $+(0,1)$  donc à 1 cm au-dessus de la rouge soit à  $(1,2)$  (point orange)



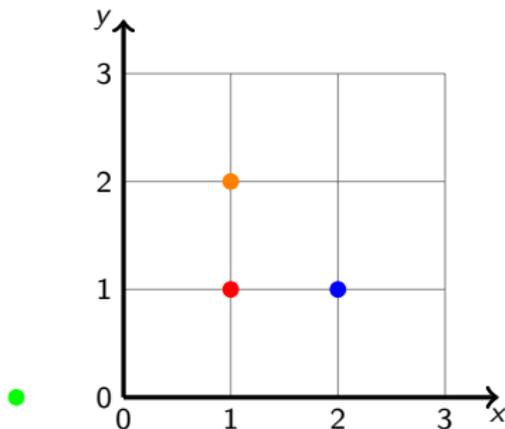
## Les coordonnées relatives

L'opérateur  $+$  se rapporte toujours à la première coordonnée définie. Si on reprend notre exemple précédent  $(1,1) + (1,0) + (-2,-1)$ , on a toujours une première coordonnée à  $(1,1)$  symbolisée par le point rouge, la 2<sup>me</sup>  $+(1,0)$  se trouve à 1 cm à gauche de la première (rouge) soit à  $(2,1)$  (point bleu), la suivante à  $+(0,1)$  donc à 1 cm au-dessus de la rouge soit à  $(1,2)$  (point orange) et enfin on a  $+(-2,-1)$  soit 2 cm vers la droite et 1 cm en bas par rapport au point rouge, donc à  $(-1,0)$  (point vert).



## Les coordonnées relatives

L'opérateur  $+$  se rapporte toujours à la première coordonnée définie. Si on reprend notre exemple précédent  $(1,1) + (1,0) + (-2,-1)$ , on a toujours une première coordonnée à  $(1,1)$  symbolisée par le point rouge, la 2<sup>me</sup>  $+(1,0)$  se trouve à 1 cm à gauche de la première (rouge) soit à  $(2,1)$  (point bleu), la suivante à  $+(0,1)$  donc à 1 cm au-dessus de la rouge soit à  $(1,2)$  (point orange) et enfin on a  $+(-2,-1)$  soit 2 cm vers la droite et 1 cm en bas par rapport au point rouge, donc à  $(-1,0)$  (point vert).

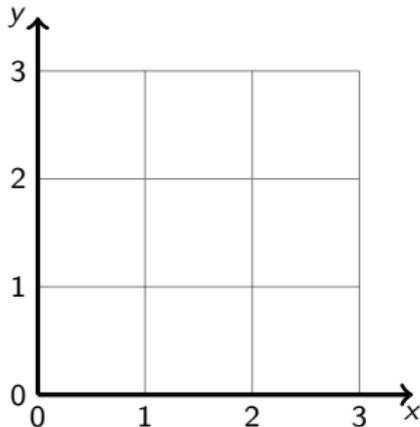


## Les chemins : path

Le principal élément de TiKZ est le chemin, `path` en anglais. Un chemin est une succession de coordonnées reliées par une opération. Par exemple `--` (2 signes moins sans espace) relie 2 coordonnées par un trait rectiligne. Le chemin suivant :

```
\path (0,0) -- (2,1) -- (2,3) -- ++(-2,-1);
```

correspond au trait rouge ci-dessous.

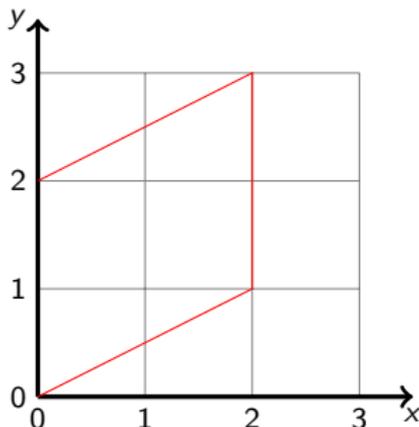


## Les chemins : path

Le principal élément de TiKZ est le chemin, `path` en anglais. Un chemin est une succession de coordonnées reliées par une opération. Par exemple `--` (2 signes moins sans espace) relie 2 coordonnées par un trait rectiligne. Le chemin suivant :

```
\path (0,0) -- (2,1) -- (2,3) -- ++(-2,-1);
```

correspond au trait rouge ci-dessous.



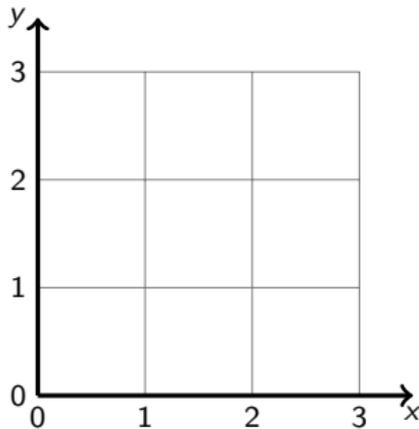
## Les opérations

Voici la liste des opérations possibles :

- - : faire un trait

-- relie 2 coordonnées par un trait :

`\path (0,0) -- (2,1)` correspond à :



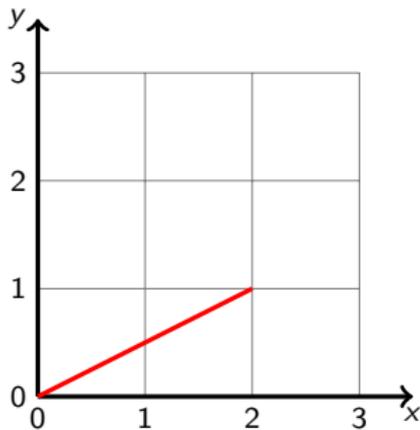
## Les opérations

Voici la liste des opérations possibles :

- - : faire un trait

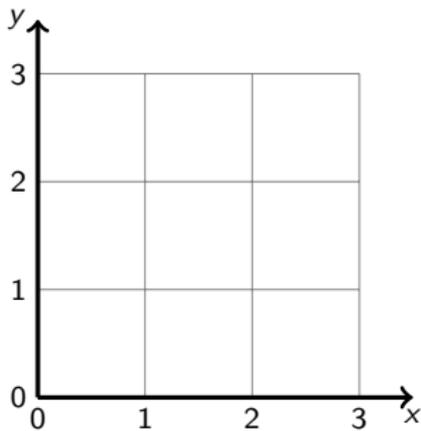
-- relie 2 coordonnées par un trait :

`\path (0,0) -- (2,1)` correspond à :



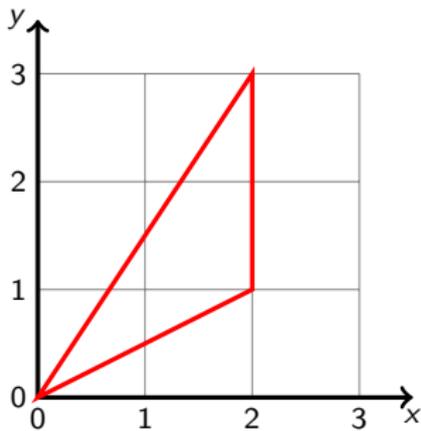
## - -cycle : ferme un chemin

On ferme un chemin avec l'option `--cycle` :  
`\path (0,0) -- (2,1) -- (2,3) --cycle;`  
correspond aux traits rouges ci-dessous.



## - -cycle : ferme un chemin

On ferme un chemin avec l'option `--cycle` :  
`\path (0,0) -- (2,1) -- (2,3) --cycle;`  
correspond aux traits rouges ci-dessous.



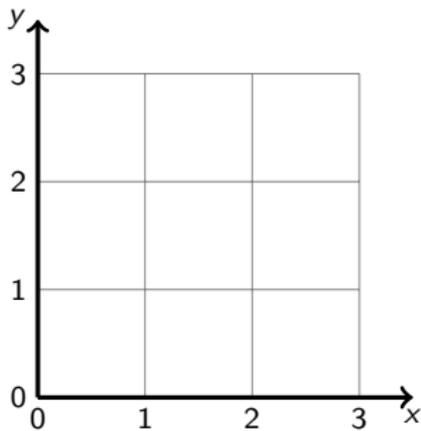
## - -cycle : ferme un chemin

On ferme un chemin avec l'option `--cycle` :

```
\path (0,0) -- (2,1) -- (2,3) --cycle;
```

correspond aux traits rouges ci-dessous.

**Attention** revenir sur l'origine de départ ne ferme pas un chemin. Le chemin suivant (en bleu) n'est pas fermée :



## - -cycle : ferme un chemin

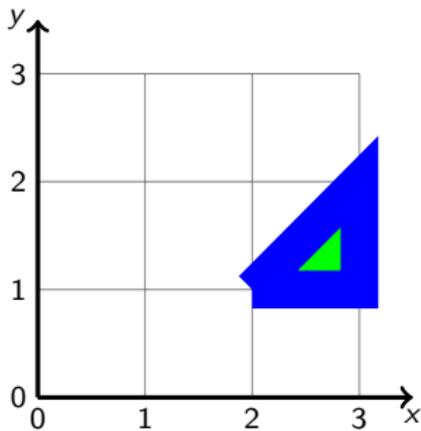
On ferme un chemin avec l'option `--cycle` :

```
\path (0,0) -- (2,1) -- (2,3) --cycle;
```

correspond aux traits rouges ci-dessous.

**Attention** revenir sur l'origine de départ ne ferme pas un chemin. Le chemin suivant (en bleu) n'est pas fermée :

```
\path (2,1) -- (3,1) -- (3,2) -- (2,1);
```



## - -cycle : ferme un chemin

On ferme un chemin avec l'option `--cycle` :

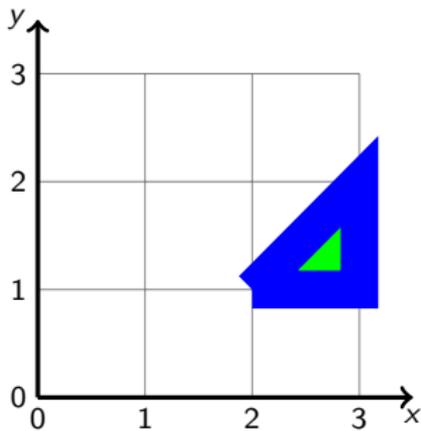
```
\path (0,0) -- (2,1) -- (2,3) --cycle;
```

correspond aux traits rouges ci-dessous.

**Attention** revenir sur l'origine de départ ne ferme pas un chemin. Le chemin suivant (en bleu) n'est pas fermée :

```
\path (2,1) -- (3,1) -- (3,2) -- (2,1);
```

Celui-ci (orange) l'est :



## - -cycle : ferme un chemin

On ferme un chemin avec l'option `--cycle` :

```
\path (0,0) -- (2,1) -- (2,3) --cycle;
```

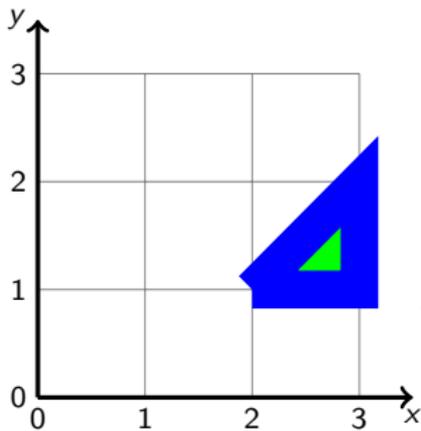
correspond aux traits rouges ci-dessous.

**Attention** revenir sur l'origine de départ ne ferme pas un chemin. Le chemin suivant (en bleu) n'est pas fermée :

```
\path (2,1) -- (3,1) -- (3,2) -- (2,1);
```

Celui-ci (orange) l'est :

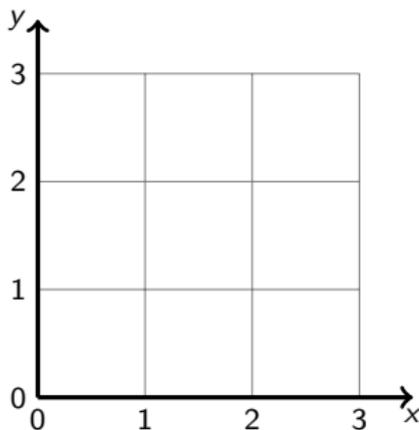
```
\path (2,1) -- (5,1) -- (5,2) --cycle;
```



## rectangle : dessine un rectangle

`rectangle` dessine un rectangle dont la première coordonnée est le coin en bas à gauche, la deuxième le coin en haut à droite :

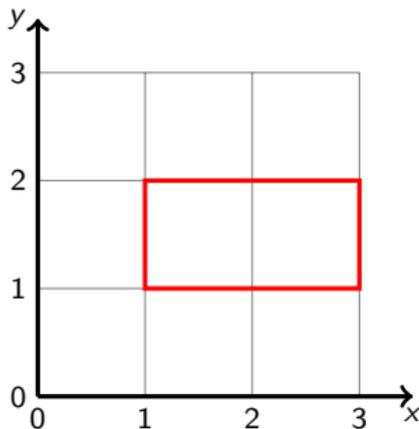
`\path (1,1) rectangle (3,2)` correspond à :



## rectangle : dessine un rectangle

`rectangle` dessine un rectangle dont la première coordonnée est le coin en bas à gauche, la deuxième le coin en haut à droite :

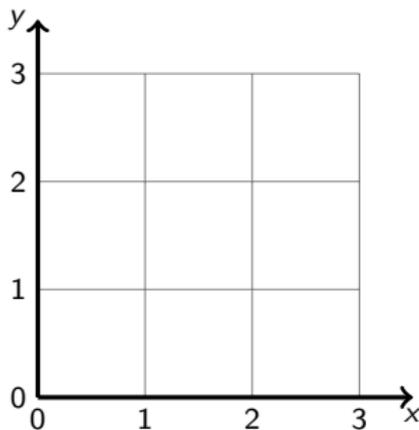
`\path (1,1) rectangle (3,2)` correspond à :



## circle : dessine un cercle

`circle` dessine un cercle dont la première coordonnée est le centre du cercle et la deuxième le rayon :

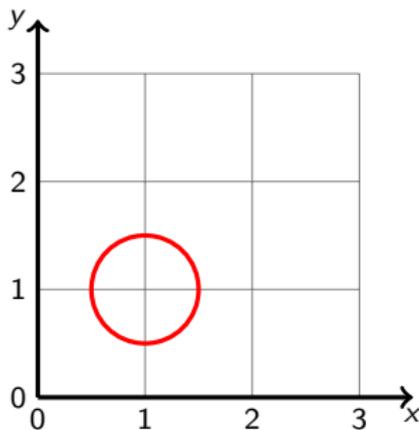
`\path (1,1) circle (5mm)` correspond à :



## circle : dessine un cercle

`circle` dessine un cercle dont la première coordonnée est le centre du cercle et la deuxième le rayon :

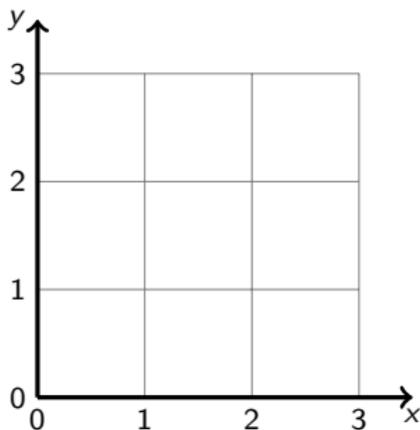
`\path (1,1) circle (5mm)` correspond à :



## ellipse : dessine une ellipse

`ellipse` dessine un cercle dont la première coordonnée est le centre du cercle et la deuxième la moitié de la largeur et la moitié de la hauteur :

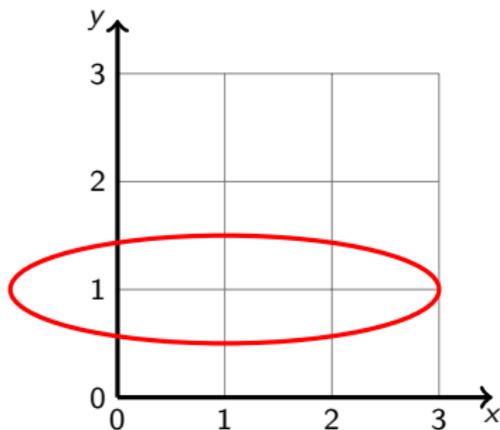
`\path (1,1) ellipse (2cm and 5mm)` correspond à :



## ellipse : dessine une ellipse

`ellipse` dessine un cercle dont la première coordonnée est le centre du cercle et la deuxième la moitié de la largeur et la moitié de la hauteur :

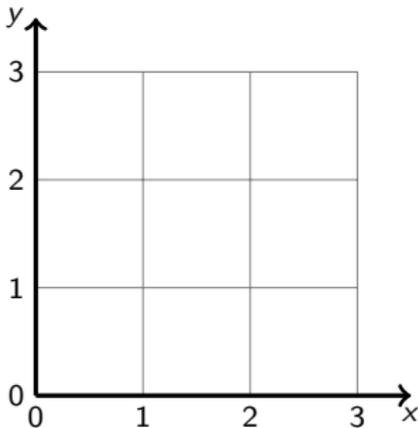
`\path (1,1) ellipse (2cm and 5mm)` correspond à :



## arc : dessine un arc de cercle

`arc` dessine un arc de cercle dont la première coordonnée est le départ de l'arc de cercle et la deuxième l'angle de départ : l'angle d'arrivé : et le rayon du cercle :

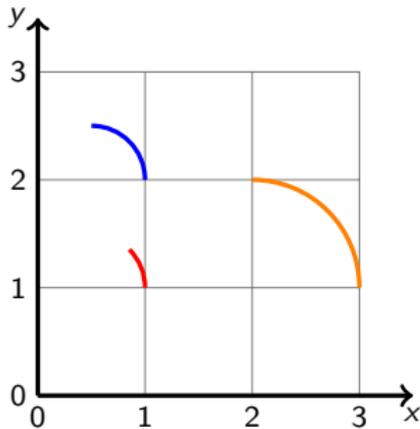
`\path (1,1) arc (0:45:5mm)` en rouge et `\path (1,2) arc (0:90:5mm)` en bleu  
`\path (3,1) arc (0:90:1cm)` en orange :



## arc : dessine un arc de cercle

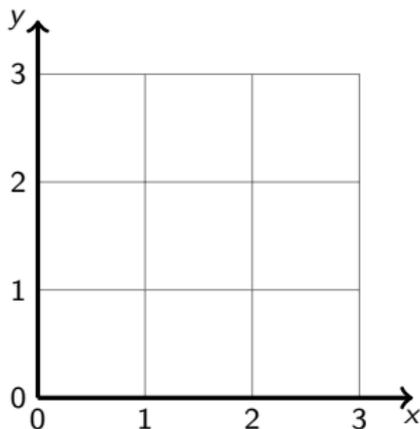
`arc` dessine un arc de cercle dont la première coordonnée est le départ de l'arc de cercle et la deuxième l'angle de départ : l'angle d'arrivé : et le rayon du cercle :

`\path (1,1) arc (0:45:5mm)` en rouge et `\path (1,2) arc (0:90:5mm)` en bleu  
`\path (3,1) arc (0:90:1cm)` en orange :



## parabola : dessine une parabole

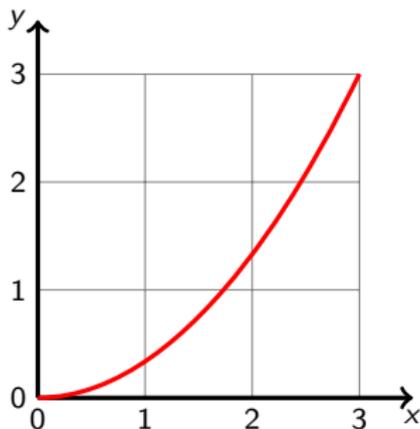
`parabola` dessine une parabole de la première coordonnée à la deuxième coordonnée. Par défaut l'inflexion de la parabole est au début  
`\path (0,0) parabola (0,3)` en rouge



## parabola : dessine une parabole

`parabola` dessine une parabole de la première coordonnée à la deuxième coordonnée. Par défaut l'inflexion de la parabole est au début

`\path (0,0) parabola (0,3)` en rouge



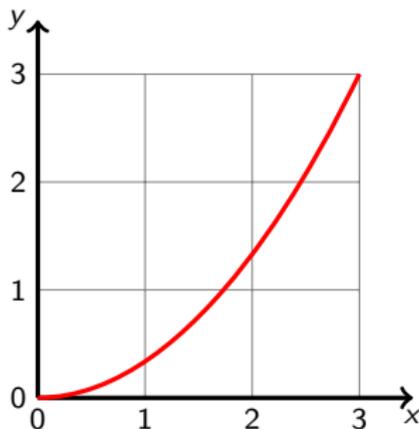
## parabola : dessine une parabole

`parabola` dessine une parabole de la première coordonnée à la deuxième coordonnée. Par défaut l'inflexion de la parabole est au début

`\path (0,0) parabola (0,3)` en rouge

Pour placer l'inflexion à la fin tu utilises `bend at end`

`\path (0,0) parabola[bend at end] (3,3)` en bleu.



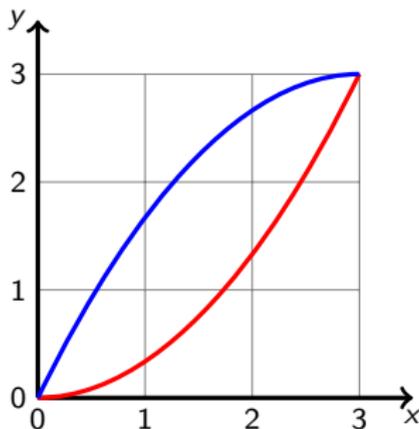
## parabola : dessine une parabole

`parabola` dessine une parabole de la première coordonnée à la deuxième coordonnée. Par défaut l'inflexion de la parabole est au début

`\path (0,0) parabola (0,3)` en rouge

Pour placer l'inflexion à la fin tu utilises `bend at end`

`\path (0,0) parabola[bend at end] (3,3)` en bleu.



## parabola : dessine une parabole

`parabola` dessine une parabole de la première coordonnée à la deuxième coordonnée. Par défaut l'inflexion de la parabole est au début

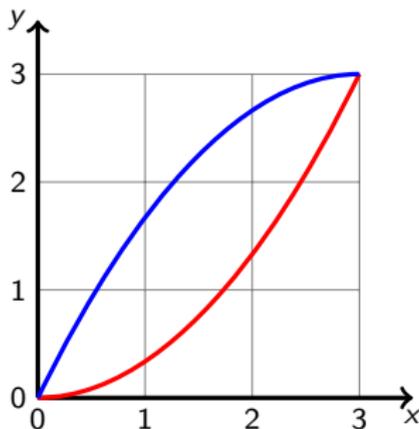
`\path (0,0) parabola (0,3)` en rouge

Pour placer l'inflexion à la fin tu utilises `bend at end`

`\path (0,0) parabola[bend at end] (3,3)` en bleu.

Enfin tu peux donner les coordonnées du point d'inflexion

`\path (0,0) parabola bend (1.5,2) (3,0)` en orange :



## parabola : dessine une parabole

`parabola` dessine une parabole de la première coordonnée à la deuxième coordonnée. Par défaut l'inflexion de la parabole est au début

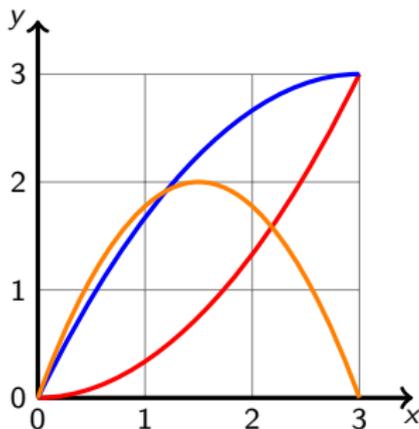
`\path (0,0) parabola (0,3)` en rouge

Pour placer l'inflexion à la fin tu utilises `bend at end`

`\path (0,0) parabola[bend at end] (3,3)` en bleu.

Enfin tu peux donner les coordonnées du point d'inflexion

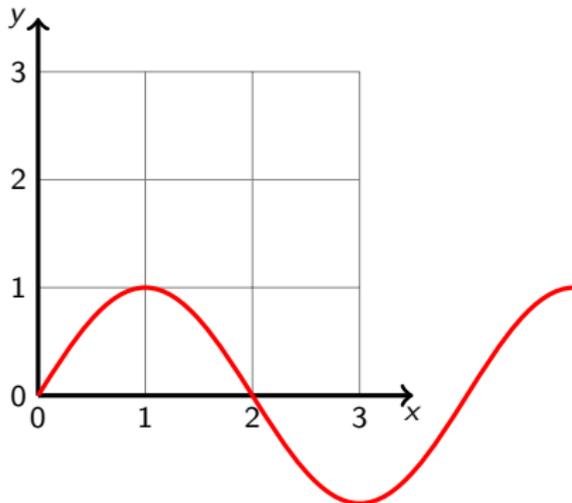
`\path (0,0) parabola bend (1.5,2) (3,0)` en orange :



## sin et cos : dessine une sinusoïde

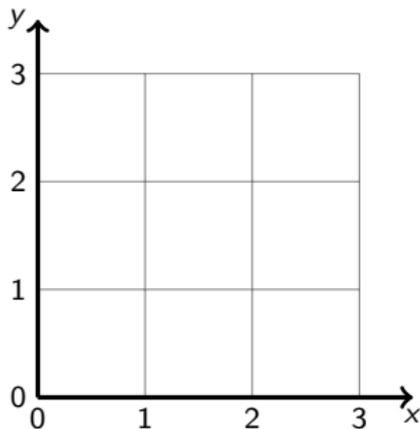
Un exemple :

```
\path (0,0) sin (1,1) cos (2,0) sin (3,-1) cos (4,0) sin (5,1);
```



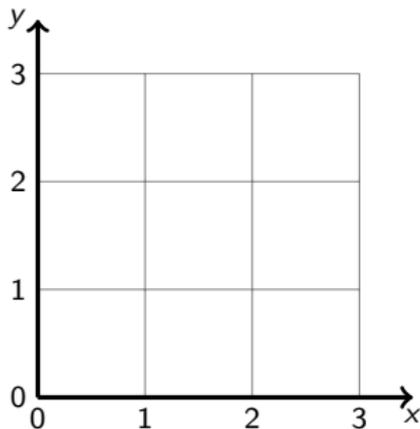
## ..controls : dessine des courbes de Bézier

Le principe est de créer un point de contrôle qui va « tirer » la ligne pour la courber.



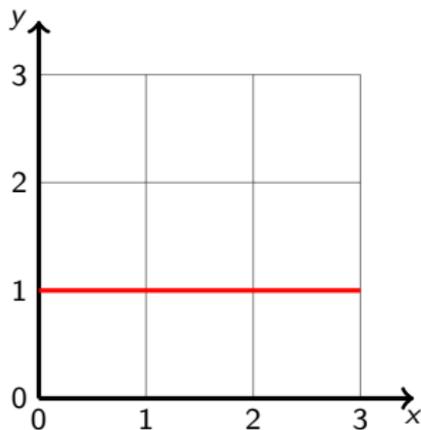
## ..controls : dessine des courbes de Bézier

Le principe est de créer un point de contrôle qui va « tirer » la ligne pour la courber. Voici une ligne rouge,



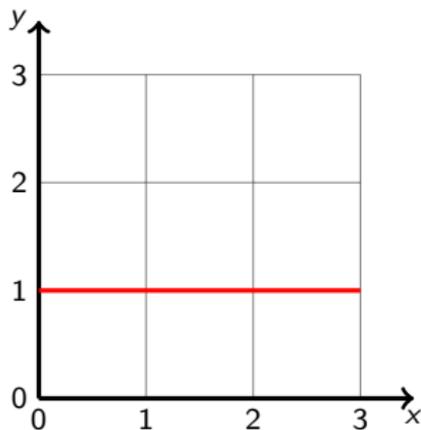
## ..controls : dessine des courbes de Bézier

Le principe est de créer un point de contrôle qui va « tirer » la ligne pour la courber. Voici une ligne rouge,



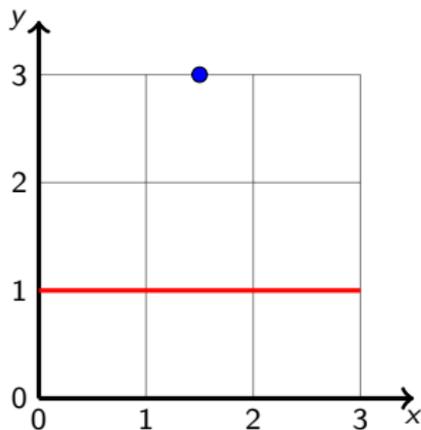
## ..controls : dessine des courbes de Bézier

Le principe est de créer un point de contrôle qui va « tirer » la ligne pour la courber. Voici une ligne rouge, un point de contrôle



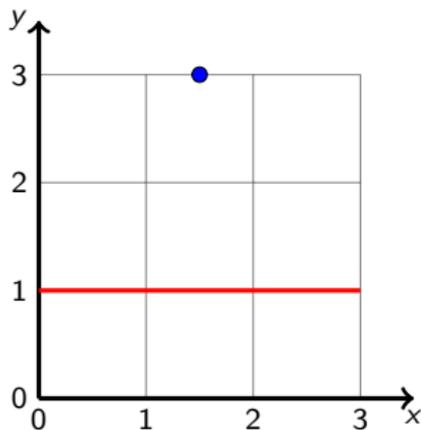
## ..controls : dessine des courbes de Bézier

Le principe est de créer un point de contrôle qui va « tirer » la ligne pour la courber. Voici une ligne rouge, un point de contrôle



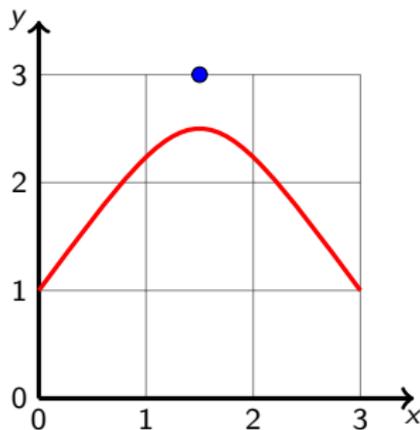
## ..controls : dessine des courbes de Bézier

Le principe est de créer un point de contrôle qui va « tirer » la ligne pour la courber. Voici une ligne rouge, un point de contrôle et la courbe résultante :



## ..controls : dessine des courbes de Bézier

Le principe est de créer un point de contrôle qui va « tirer » la ligne pour la courber. Voici une ligne rouge, un point de contrôle et la courbe résultante :



## ..controls : dessine des courbes de Bézier

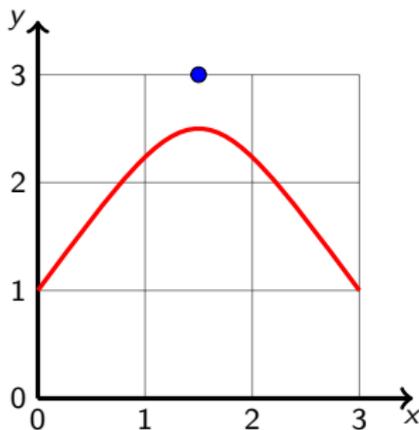
Le principe est de créer un point de contrôle qui va « tirer » la ligne pour la courber. Voici une ligne rouge, un point de contrôle et la courbe résultante :

## La commande

Un point de contrôle s'écrit `.. controls (x,y) ..`. Attention il s'agit de deux points et non de tirets et controls prend un s

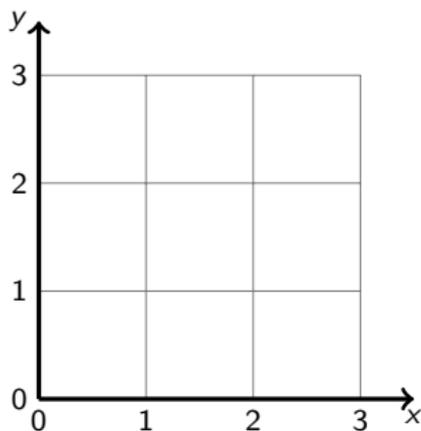
Voici la commande pour créer la courbe :

```
\path (0,1) ..controls (1.5,3) .. (3,1);
```



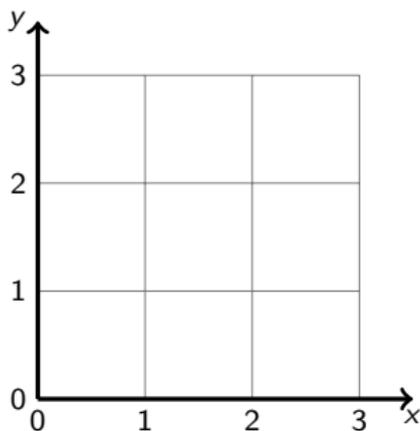
## Les courbes de Bézier

Tu peux placer plusieurs points de contrôle à la suite. Il suffit de les séparer par `and`.



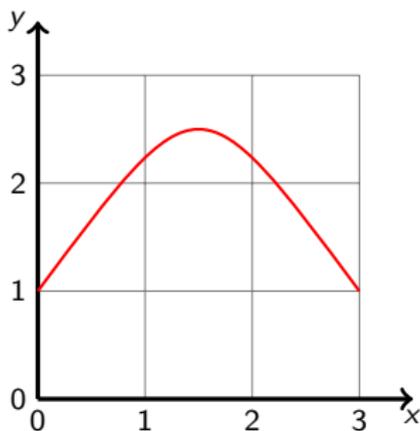
## Les courbes de Bézier

Tu peux placer plusieurs points de contrôle à la suite. Il suffit de les séparer par `and`. On reprend notre exemple précédent



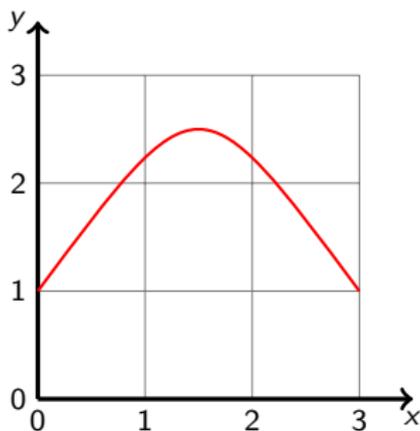
## Les courbes de Bézier

Tu peux placer plusieurs points de contrôle à la suite. Il suffit de les séparer par `and`. On reprend notre exemple précédent



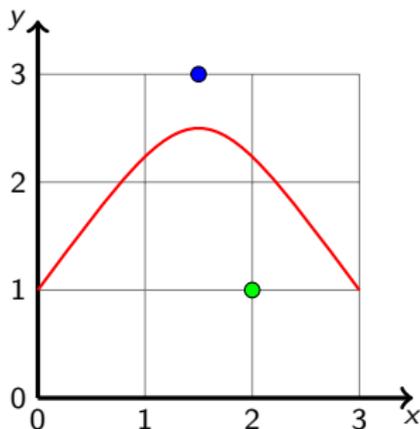
## Les courbes de Bézier

Tu peux placer plusieurs points de contrôle à la suite. Il suffit de les séparer par `and`. On reprend notre exemple précédent et l'on va ajouter un nouveau point de contrôle (en vert)



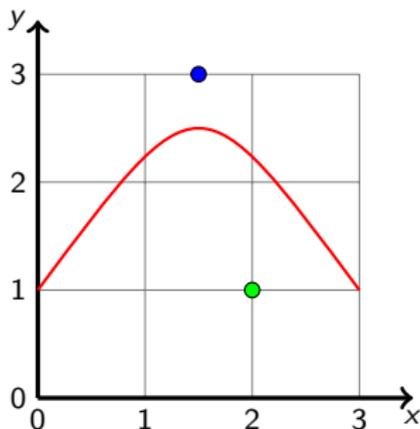
## Les courbes de Bézier

Tu peux placer plusieurs points de contrôle à la suite. Il suffit de les séparer par `and`. On reprend notre exemple précédent et l'on va ajouter un nouveau point de contrôle (en vert)



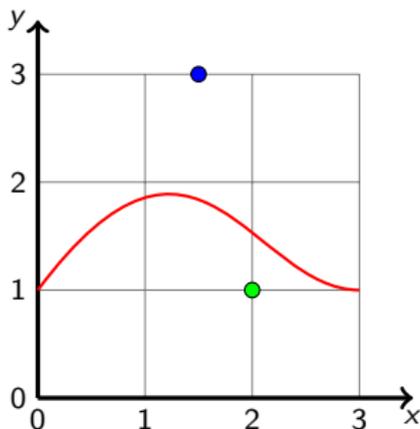
## Les courbes de Bézier

Tu peux placer plusieurs points de contrôle à la suite. Il suffit de les séparer par `and`. On reprend notre exemple précédent et l'on va ajouter un nouveau point de contrôle (en vert) et la courbe résultante :



## Les courbes de Bézier

Tu peux placer plusieurs points de contrôle à la suite. Il suffit de les séparer par `and`. On reprend notre exemple précédent et l'on va ajouter un nouveau point de contrôle (en vert) et la courbe résultante :

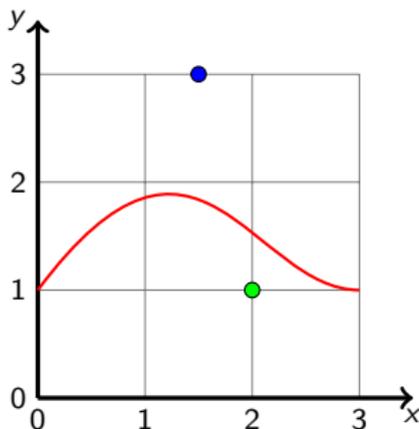


## Les courbes de Bézier

Tu peux placer plusieurs points de contrôle à la suite. Il suffit de les séparer par `and`. On reprend notre exemple précédent et l'on va ajouter un nouveau point de contrôle (en vert) et la courbe résultante :

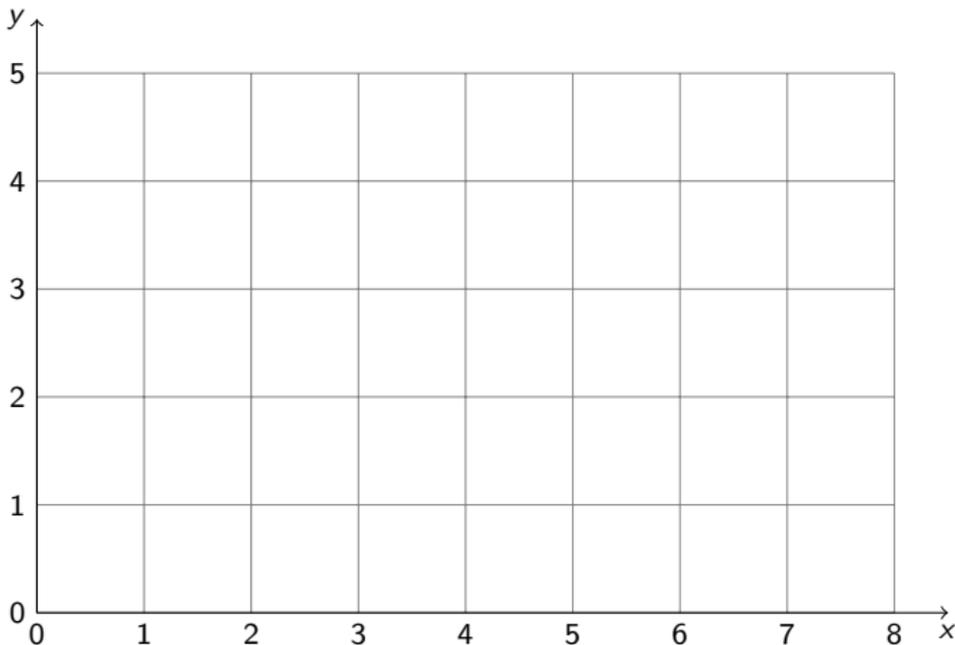
## La commande

```
\path (0,1) ..controls (1.5,3) and (1,2) .. (3,1);
```



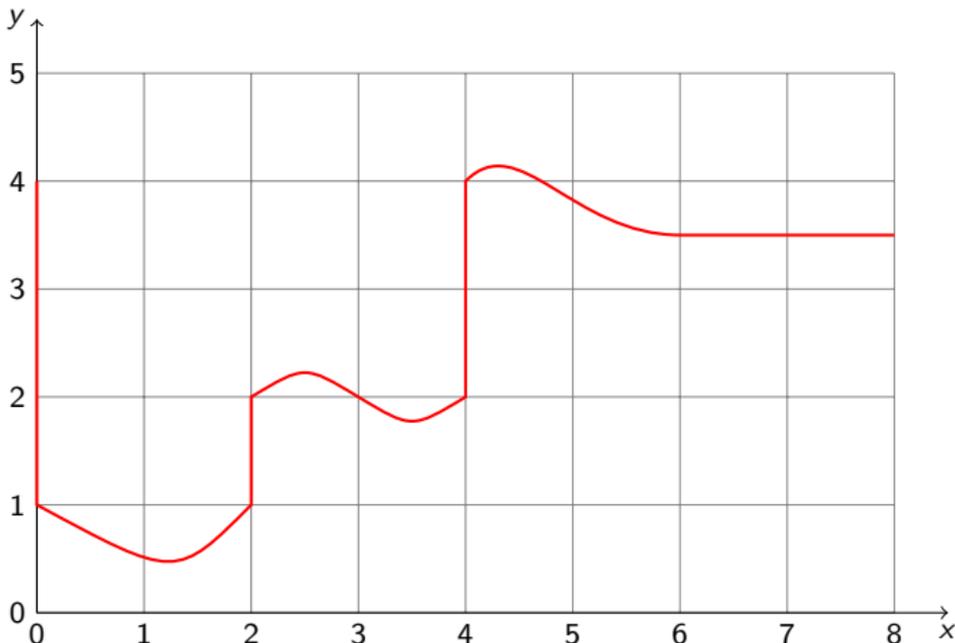
## Les courbes de Bézier

Dessiner des figures complexes n'est pas évident. Voici comment je procède pour obtenir la courbe rouge suivante.



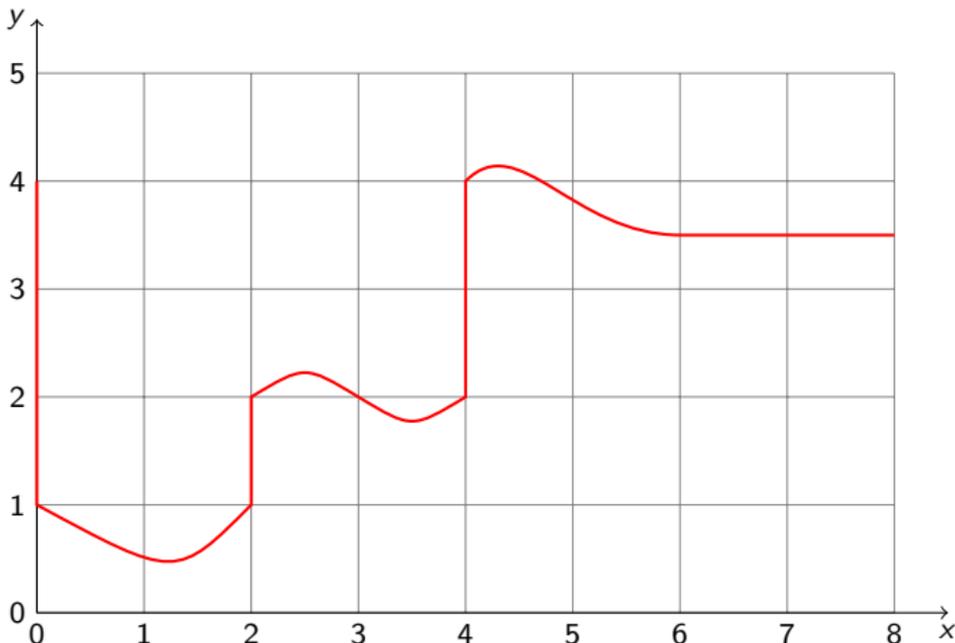
## Les courbes de Bézier

Dessiner des figures complexes n'est pas évident. Voici comment je procède pour obtenir la courbe rouge suivante.



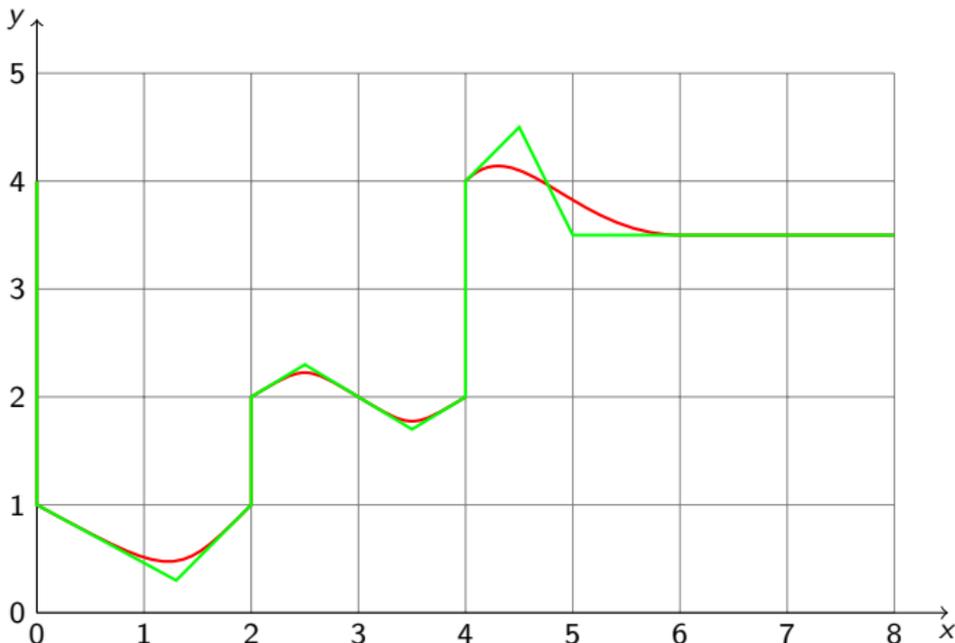
## Les courbes de Bézier

Dessiner des figures complexes n'est pas évident. Voici comment je procède pour obtenir la courbe rouge suivante. Je trace d'abord (sur un papier quadrillé) une courbe (ici en vert) composée de segments de droite au plus près de la courbe désirée.



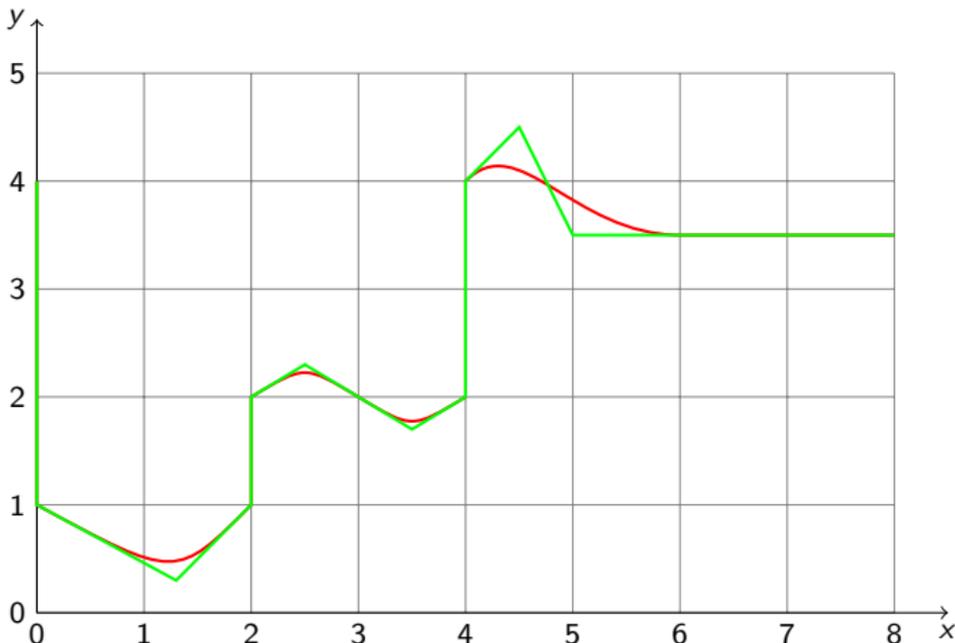
## Les courbes de Bézier

Dessiner des figures complexes n'est pas évident. Voici comment je procède pour obtenir la courbe rouge suivante. Je trace d'abord (sur un papier quadrillé) une courbe (ici en vert) composée de segments de droite au plus près de la courbe désirée.



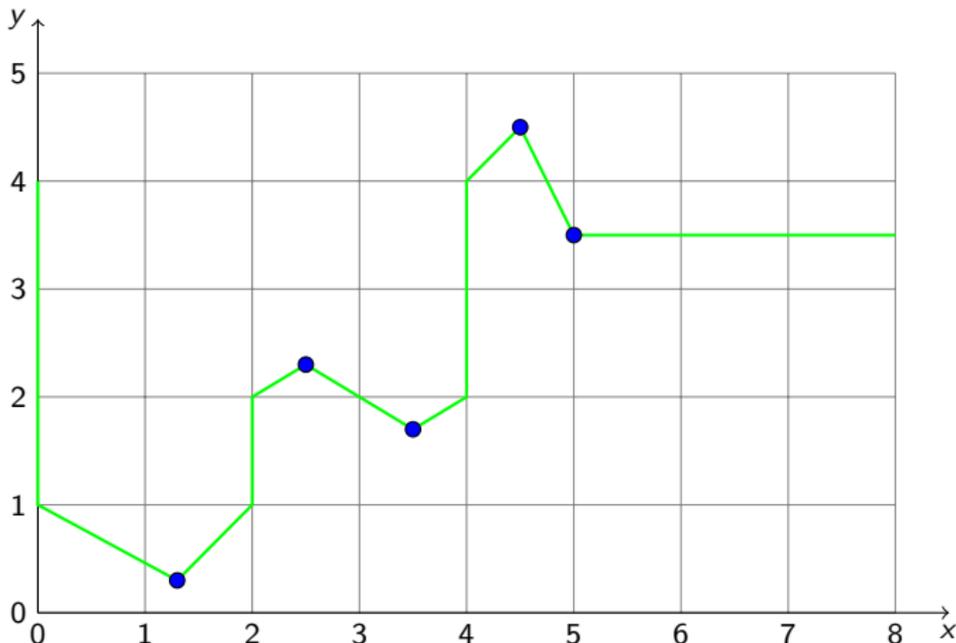
## Les courbes de Bézier

Dessiner des figures complexes n'est pas évident. Voici comment je procède pour obtenir la courbe rouge suivante. Je trace d'abord (sur un papier quadrillé) une courbe (ici en vert) composée de segments de droite au plus près de la courbe désirée. Puis je transforme les points en bleu en point de contrôle.



## Les courbes de Bézier

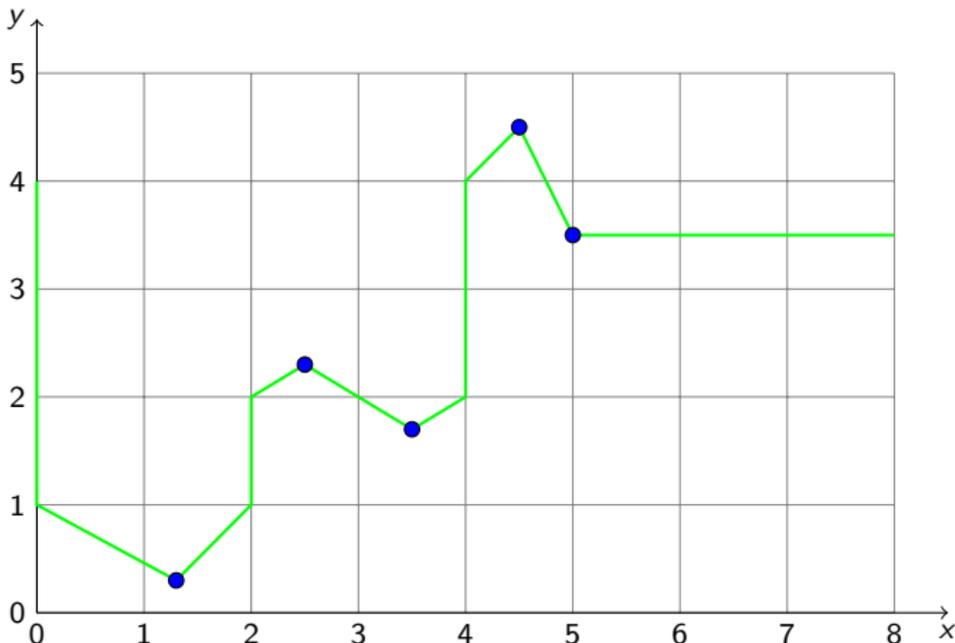
Dessiner des figures complexes n'est pas évident. Voici comment je procède pour obtenir la courbe rouge suivante. Je trace d'abord (sur un papier quadrillé) une courbe (ici en vert) composée de segments de droite au plus près de la courbe désirée. Puis je transforme les points en bleu en point de contrôle.



## Les courbes de Bézier

Dessiner des figures complexes n'est pas évident. Voici comment je procède pour obtenir la courbe rouge suivante. Je trace d'abord (sur un papier quadrillé) une courbe (ici en vert) composée de segments de droite au plus près de la courbe désirée. Puis je transforme les points en bleu en point de contrôle. J'obtiens la commande suivante :

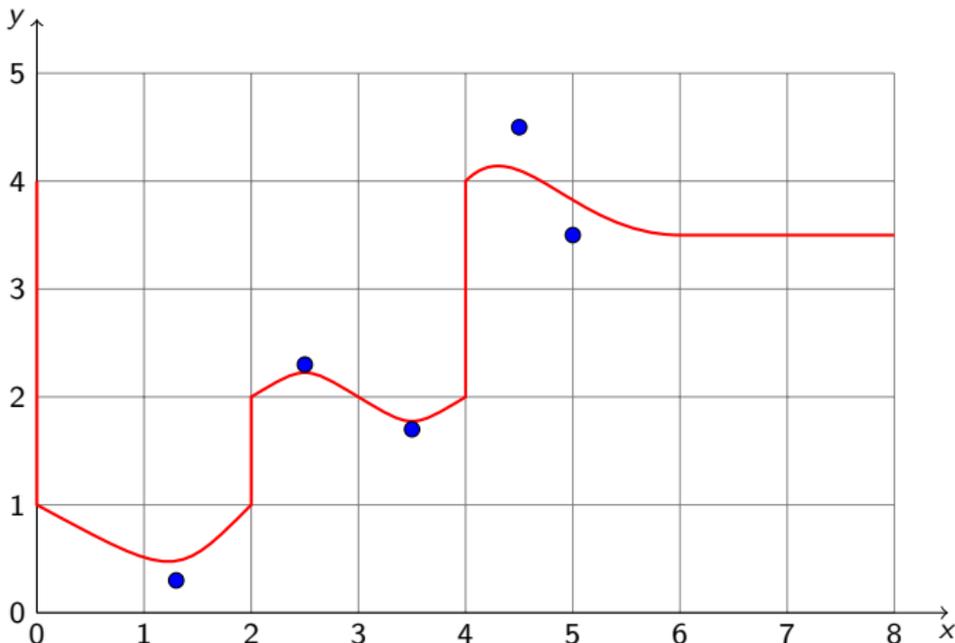
```
\path (0,4) -- (0,1) ..controls (1.3,0.3)..(2,1)--(2,2)
..controls(2.5,2.3)..(3,2)..controls (3.5,1.7) ..(4,2)--(4,4)
..controls(4.5,4.5) and (5,3.5) .. (6,3.5)--(8,3.5);
```



## Les courbes de Bézier

Dessiner des figures complexes n'est pas évident. Voici comment je procède pour obtenir la courbe rouge suivante. Je trace d'abord (sur un papier quadrillé) une courbe (ici en vert) composée de segments de droite au plus près de la courbe désirée. Puis je transforme les points en bleu en point de contrôle. J'obtiens la commande suivante :

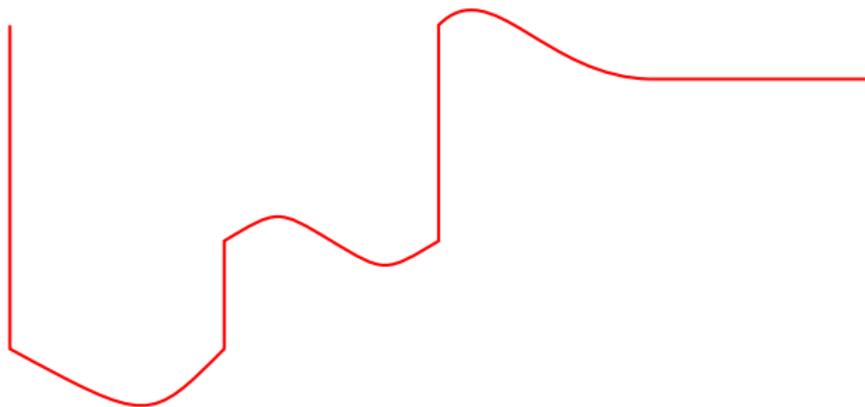
```
\path (0,4) -- (0,1) ..controls (1.3,0.3)..(2,1)--(2,2)
  ..controls(2.5,2.3)..(3,2)..controls (3.5,1.7) .. (4,2)--(4,4)
  ..controls(4.5,4.5) and (5,3.5) .. (6,3.5)--(8,3.5);
```



## Les courbes de Bézier

Dessiner des figures complexes n'est pas évident. Voici comment je procède pour obtenir la courbe rouge suivante. Je trace d'abord (sur un papier quadrillé) une courbe (ici en vert) composée de segments de droite au plus près de la courbe désirée. Puis je transforme les points en bleu en point de contrôle. J'obtiens la commande suivante :

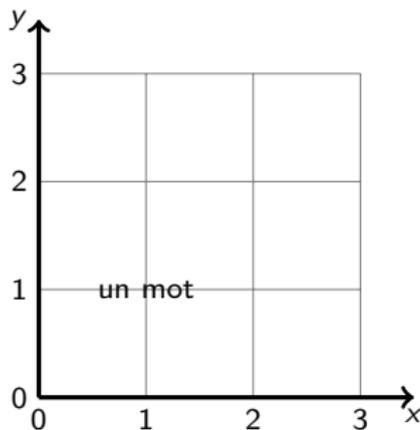
```
\path (0,4) -- (0,1) ..controls (1.3,0.3)..(2,1)--(2,2)  
  ..controls(2.5,2.3)..(3,2)..controls (3.5,1.7) ..(4,2)--(4,4)  
  ..controls(4.5,4.5) and (5,3.5) .. (6,3.5)--(8,3.5);
```



## node : insérer du texte

`node` permet d'insérer du texte, ou tout élément  $\LaTeX$  (tableau, image, minipage, liste. . .) centré au point de coordonnée précisé.

```
\path (1,1) node{un mot};
```



## grid : dessine une grille

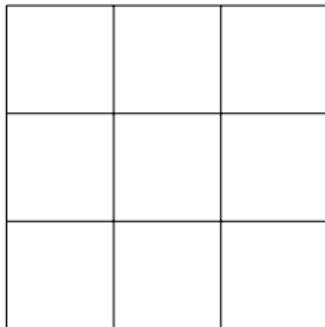
`grid` dessine une grille dont la première coordonnée est le coin en bas à gauche, la deuxième le coin en haut à droite. Voici une grille de 3 cm de côté :

```
\tikz \draw (0,0) grid (3,3);
```

## grid : dessine une grille

`grid` dessine une grille dont la première coordonnée est le coin en bas à gauche, la deuxième le coin en haut à droite. Voici une grille de 3 cm de côté :

```
\tikz \draw (0,0) grid (3,3);
```



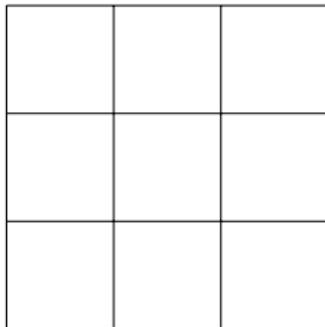
## grid : dessine une grille

`grid` dessine une grille dont la première coordonnée est le coin en bas à gauche, la deuxième le coin en haut à droite. Voici une grille de 3 cm de côté :

```
\tikz \draw (0,0) grid (3,3);
```

La commande `\grid` accepte des options qui comme en  $\text{\LaTeX}$  sont notées entre crochets `[ ]`, il s'agit de `step` qui précise le pas de la grille . Par exemple pour une grille de 3 cm de côté et dessinée tous les 5 mm

```
\tikz \draw[step=5mm] (0,0) grid (3,3);
```



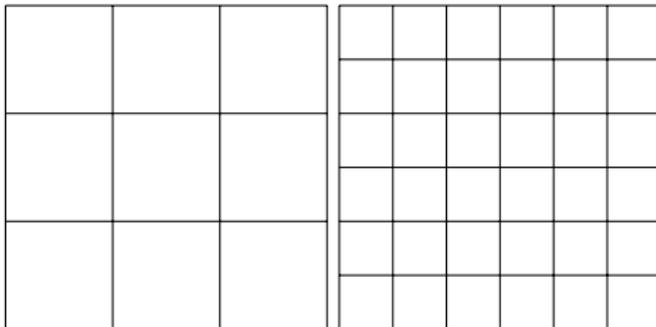
## grid : dessine une grille

`grid` dessine une grille dont la première coordonnée est le coin en bas à gauche, la deuxième le coin en haut à droite. Voici une grille de 3 cm de côté :

```
\tikz \draw (0,0) grid (3,3);
```

La commande `\grid` accepte des options qui comme en  $\text{\LaTeX}$  sont notées entre crochets `[ ]`, il s'agit de `step` qui précise le pas de la grille . Par exemple pour une grille de 3 cm de côté et dessinée tous les 5 mm

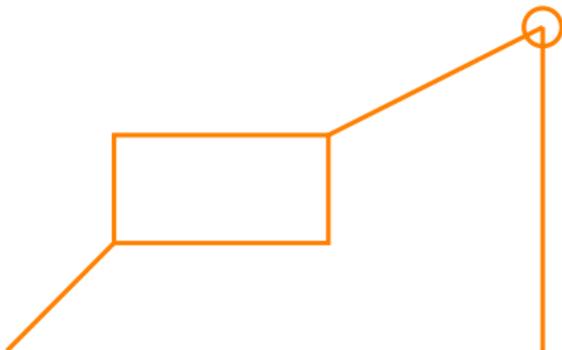
```
\tikz \draw[step=5mm] (0,0) grid (3,3);
```



## Les chemins : path

On peut mélanger les opérations. Voici un chemin qui mélange des traits, un rectangle et un cercle :

```
\path (0,0) -- (1,1) rectangle (3,2) -- (5,3) circle (5pt) -- (5,0);
```



Introduction

Le système de  
coordonnées

Les coordonnées  
relatives

Les coordonnées  
relatives

Les chemins : path

Les opérations

**Les actions**

La couleur

Position des options

Les actions (suite)

conclusion

## Les actions

Par défaut la commande `\path`, qui définit un chemin ne fait rien avec celui-ci. Tu dois dire ce que tu veux faire de ce chemin. Tu peux le dessiner, le remplir de couleur, lui appliquer une trame. . .

Tu peux lui attribuer une couleur, une épaisseur de trait, déterminer la forme des angles. . .

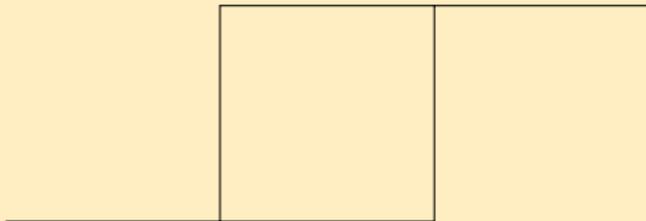
Comme toutes les options  $\LaTeX$  elles sont encadrées de `[ ]`.

Les pages suivantes montre la listes des actions possibles.

## draw : dessine

`draw`, permet de tracer le contour d'un chemin, comme si tu utilisais un stylo.

```
\tikz \path [draw] (0,0)-- (2,0) rectangle (4,2) -- (6,2);
```



`\path [draw]` c'est trop long à écrire donc Till Tantau qui est un mec sympa a créé des raccourcis. `\draw` est égal à `\path [draw]`. La figure précédente peut s'écrire :

```
\tikz \draw (0,0)-- (2,0) rectangle (4,2) -- (6,2);
```

## La couleur

Avant de continuer la liste des actions on va faire un aparté pour évoquer la couleur.

La couleur sous TiKZ est gérée par le package `xcolor` qui est chargé automatiquement. Toutes les commandes vues dans la fiche «  $\text{\LaTeX}$  & la couleur » sont utilisables sous TiKZ. Il y a plusieurs façons d'appliquer la couleur. Si l'on reprend la commande `draw`, pour dessiner en rouge toutes les commandes suivantes ont le même effet.

```
\path[draw,color=red]
```

```
\path[draw=red]
```

```
\draw[color=red]
```

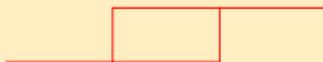
```
\draw[red]
```

La couleur par défaut est le noir, `draw` sans option dessine un trait noir.

## draw & la couleur

Donc pour avoir un trait rouge, il suffit d'écrire :

```
\tikz \draw [draw=red](0,0)-- (2,0) rectangle (4,0.5) -- (6,0.5);
```

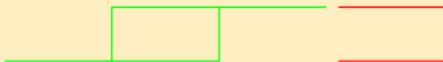


**Attention** tu ne peux pas changer de couleur dans un même chemin. Pour mettre le trait en rouge et le rectangle en vert tu ne peux pas faire :

```
\tikz \draw [draw=red](0,0)--(2,0) [draw=green]rectangle(2,0.5)--(3,0.5);
```

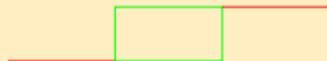
De même pour 2 traits de couleurs différentes la commande suivante ne marche pas :

```
\tikz \draw [draw=green](0,0)-- (1,0) [draw=red] (0,0.5)--(1,0.5);
```

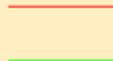


C'est la dernière couleur indiquée qui l'emporte. Pour obtenir les résultats escomptés il faut écrire :

```
\begin{tikzpicture}
  \draw [draw=red](0,0)-- (1,0);
  \draw [draw=green] (1,0) rectangle (2,0.5);
  \draw [draw=red](2,0.5) -- (3,0.5);
\end{tikzpicture}
```



```
\begin{tikzpicture}
  \draw [draw=green](0,0)-- (1,0);
  \draw [draw=red](0,.5) -- (1,.5);
\end{tikzpicture}
```



## Position des options

Deuxième aparté : la position des options dans la ligne de commande TikZ.

Tu peux les mettre où tu veux. Les 4 commandes ci-dessous donne le même résultat. `\tikz \path [draw] (0,0) rectangle (4,1);`

`\tikz \path (0,0) [draw] rectangle (4,1);`

`\tikz \path (0,0) rectangle [draw] (4,1);`

`\tikz \path (0,0) rectangle (4,1) [draw];`

Une autre remarque, les espaces ne sont pas utiles (sauf pour la compréhension du code). Tu peux écrire :

`\tikz\path[draw](0,0)rectangle(4,1);`

Maintenant on revient aux actions.

## fill : rempli de couleur

`fill`, permet de remplir un chemin de couleur comme si tu utilisais un pinceau.

```
\tikz \path [fill=orange] (0,0) rectangle (1,1) (2,1) circle (5pt);
```



Le raccourci : `\fill = \path[fill]`.

`fill` ne dessine pas le contour. Pour ce faire tu dois utiliser l'option `draw` :

```
\tikz \fill [fill=orange,draw=blue] (0,0) rectangle (1,1) (2,1) circle (5pt);
```



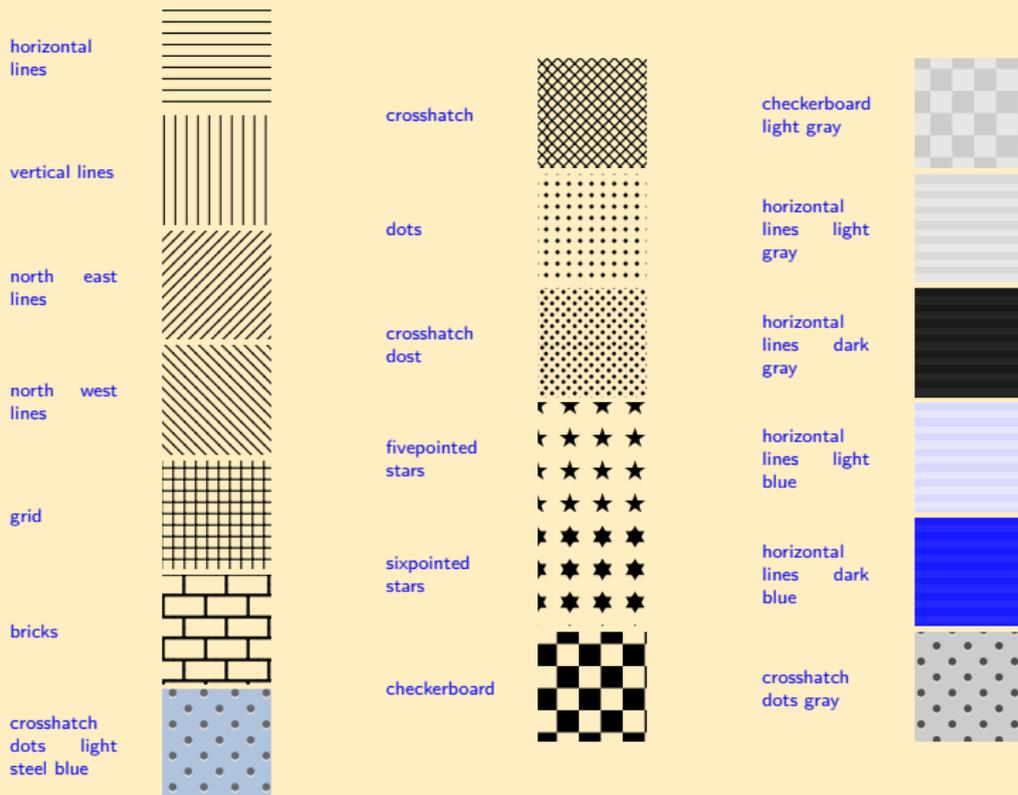
Pour dessiner à la fois le remplissage et les traits, il existe un raccourci : `\filldraw` :

```
\tikz \filldraw [red] (0,0)-- (1,0) rectangle (2,.5) -- (3,.5);
```



## Appliquer une trame : pattern

La commande `pattern` applique une trame au chemin. Il est nécessaire de charger la bibliothèque de trame pour l'utiliser `\usetikzlibrary{patterns}` à placer dans l'entête de ton source. Voici la liste des trames disponibles :



## Appliquer une trame : pattern

Tu utilises `pattern` de cette façon :

```
\tikz \path [pattern=fivepointed stars](0,0)rectangle (3,1);
```

Attention de ne pas oublier les espaces dans le nom des trames



Comme pour `fill`, `pattern` ne dessine pas le contour du chemin. Pour ce faire il faut utiliser :

```
\tikz \draw [pattern=fivepointed stars](0,0)rectangle (3,1);
```

ou

```
\tikz \path [pattern=fivepointed stars,draw](0,0)rectangle (3,1);
```



Il existe une commande `\pattern`

```
\tikz \pattern [pattern=fivepointed stars,draw](0,0)rectangle (3,1);
```

donne le même résultat que ci-dessus.

## Appliquer une trame : pattern

Tu peux changer la couleur des trames :

```
\tikz \draw [pattern=fivepointed stars,pattern color=red](0,0)rectangle (3,1);
```



Changer le fond : c'est un peu plus compliqué, il faut avoir vu d'autres notions, donc on verra cela plus tard, dans une autre fiche.

## Appliquer un dégradé : shade

`shade` permet d'appliquer un dégradé à un chemin. Elle fonctionne comme `fill`. Il existe un raccourci `\shade = \path[shade]`. Elle ne dessine pas le contour, donc il existe un `\shadedraw`.

## Appliquer un dégradé : shade

`shade` permet d'appliquer un dégradé à un chemin. Elle fonctionne comme `fill`. Il existe un raccourci `\shade = \path[shade]`. Elle ne dessine pas le contour, donc il existe un `\shadedraw`. Voici ce que donne la commande suivante :  
`\tikz \shade (0,10pt) circle (10pt) (2,0) rectangle (5,1);`

## Appliquer un dégradé : shade

`shade` permet d'appliquer un dégradé à un chemin. Elle fonctionne comme `fill`. Il existe un raccourci `\shade = \path[shade]`. Elle ne dessine pas le contour, donc il existe un `\shadedraw`. Voici ce que donne la commande suivante :  
`\tikz \shade (0,10pt) circle (10pt) (2,0) rectangle (5,1);`



## Appliquer un dégradé : shade

`shade` permet d'appliquer un dégradé à un chemin. Elle fonctionne comme `fill`. Il existe un raccourci `\shade = \path[shade]`. Elle ne dessine pas le contour, donc il existe un `\shadedraw`. Voici ce que donne la commande suivante :

```
\tikz \shade (0,10pt) circle (10pt) (2,0) rectangle (5,1);
```

Et la même avec un contour :

```
\tikz \shadedraw (0,10pt) circle (10pt) (2,0) rectangle (5,1);
```



## Appliquer un dégradé : shade

`shade` permet d'appliquer un dégradé à un chemin. Elle fonctionne comme `fill`. Il existe un raccourci `\shade = \path[shade]`. Elle ne dessine pas le contour, donc il existe un `\shadedraw`. Voici ce que donne la commande suivante :

```
\tikz \shade (0,10pt) circle (10pt) (2,0) rectangle (5,1);
```

Et la même avec un contour :

```
\tikz \shadedraw (0,10pt) circle (10pt) (2,0) rectangle (5,1);
```



Introduction

Le système de  
coordonnées

Les coordonnées  
relatives

Les coordonnées  
relatives

Les chemins : path

Les opérations

Les actions

La couleur

Position des options

Les actions (suite)

conclusion

## Appliquer un dégradé : shade

Il existe 3 types de dégradé :

## Appliquer un dégradé : shade

Il existe 3 types de dégradé :

axis le dégradé par défaut, de haut en bas

```
\tikz \shadedraw [shading=axis] (0,10pt) circle (10pt);
```

```
\tikz \shadedraw [shading=axis] (0,0) rectangle (3,1);
```



## Appliquer un dégradé : shade

Il existe 3 types de dégradé :

**axis** le dégradé par défaut, de haut en bas

```
\tikz \shadedraw [shading=axis] (0,10pt) circle (10pt);
```

```
\tikz \shadedraw [shading=axis] (0,0) rectangle (3,1);
```

**radial** du centre vers les bords

```
\tikz \shadedraw [shading=radial] (0,10pt) circle (10pt) (2,0);
```

```
\tikz \shadedraw [shading=radial] (0,0) rectangle (3,1);
```



## Appliquer un dégradé : shade

Il existe 3 types de dégradé :

**axis** le dégradé par défaut, de haut en bas

```
\tikz \shadedraw [shading=axis] (0,10pt) circle (10pt);
```

```
\tikz \shadedraw [shading=axis] (0,0) rectangle (3,1);
```

**radial** du centre vers les bords

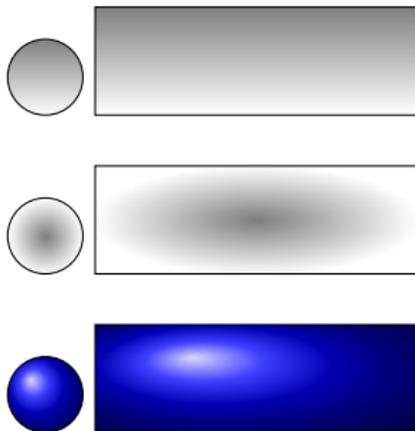
```
\tikz \shadedraw [shading=radial] (0,10pt) circle (10pt) (2,0);
```

```
\tikz \shadedraw [shading=radial] (0,0) rectangle (3,1);
```

**ball** comme une balle avec une couleur bleu par défaut

```
\tikz \shadedraw [shading=ball] (0,10pt) circle (10pt);
```

```
\tikz \shadedraw [shading=ball] (0,0) rectangle (3,1);
```



## Appliquer un dégradé : shade

Tu peux appliquer une rotation au dégradé (**Attention pas au chemin**) avec l'option `shading angle=angle en degrés`. Voici quelques exemples :

## Appliquer un dégradé : shade

Tu peux appliquer une rotation au dégradé (**Attention pas au chemin**) avec l'option `shading angle=angle en degrés`. Voici quelques exemples :

```
\tikz \shadedraw [shading=axis,shading angle=45] (0,0) rectangle (3,1);
```



## Appliquer un dégradé : shade

Tu peux appliquer une rotation au dégradé (**Attention pas au chemin**) avec l'option `shading angle=angle en degrés`. Voici quelques exemples :

```
\tikz \shadedraw [shading=axis,shading angle=45] (0,0) rectangle (3,1);  
\tikz \shadedraw [shading=axis,shading angle=90] (0,0) rectangle (3,1);
```



## Appliquer un dégradé : shade

Tu peux appliquer une rotation au dégradé (**Attention pas au chemin**) avec l'option `shading angle=angle en degrés`. Voici quelques exemples :

```
\tikz \shadedraw [shading=axis,shading angle=45] (0,0) rectangle (3,1);  
\tikz \shadedraw [shading=axis,shading angle=90] (0,0) rectangle (3,1);  
\tikz \shadedraw [shading=ball,shading angle=90] (0,10pt) circle (10pt);
```



## Appliquer un dégradé : shade

Tu peux appliquer une rotation au dégradé (**Attention pas au chemin**) avec l'option `shading angle=angle en degrés`. Voici quelques exemples :

```
\tikz \shadedraw [shading=axis,shading angle=45] (0,0) rectangle (3,1);
\tikz \shadedraw [shading=axis,shading angle=90] (0,0) rectangle (3,1);
\tikz \shadedraw [shading=ball,shading angle=90] (0,10pt) circle (10pt);
\tikz \shadedraw [shading=ball,shading angle=45] (0,10pt) circle (10pt);
```



## La couleur d'un dégradé

Changer la couleur d'un dégradé est un peu compliqué et dépend du type de dégradé. On va étudier tout d'abord le dégradé axial. Il y a deux commandes pour le faire : `top color=couleur` et `bottom color=couleur`. Quand tu utilises ces options tu passes automatiquement en mode `shade` avec `shading=axis` et `shade angle=0`, donc pas besoin de les préciser. Tu peux toutefois changer l'angle. Par défaut `top color=gris` et `bottom color=blanc`. Exemples :

## La couleur d'un dégradé

Changer la couleur d'un dégradé est un peu compliqué et dépend du type de dégradé. On va étudier tout d'abord le dégradé axial. Il y a deux commandes pour le faire : `top color=couleur` et `bottom color=couleur`. Quand tu utilises ces options tu passes automatiquement en mode `shade` avec `shading=axis` et `shade angle=0`, donc pas besoin de les préciser. Tu peux toutefois changer l'angle. Par défaut `top color=gris` et `bottom color=blanc`. Exemples :

```
\tikz \draw [top color=orange] (0,0) rectangle (3,1);
```



## La couleur d'un dégradé

Changer la couleur d'un dégradé est un peu compliqué et dépend du type de dégradé. On va étudier tout d'abord le dégradé axial. Il y a deux commandes pour le faire : `top color=couleur` et `bottom color=couleur`. Quand tu utilises ces options tu passes automatiquement en mode `shade` avec `shading=axis` et `shade angle=0`, donc pas besoin de les préciser. Tu peux toutefois changer l'angle. Par défaut `top color=gris` et `bottom color=blanc`. Exemples :

```
\tikz \draw [top color=orange] (0,0) rectangle (3,1);
```



```
\tikz \draw [top color=orange,shading angle=45] (0,0) rectangle (3,1);
```



## La couleur d'un dégradé

Changer la couleur d'un dégradé est un peu compliqué et dépend du type de dégradé. On va étudier tout d'abord le dégradé axial. Il y a deux commandes pour le faire : `top color=couleur` et `bottom color=couleur`. Quand tu utilises ces options tu passes automatiquement en mode `shade` avec `shading=axis` et `shade angle=0`, donc pas besoin de les préciser. Tu peux toutefois changer l'angle. Par défaut `top color=gris` et `bottom color=blanc`. Exemples :

```
\tikz \draw [top color=orange] (0,0) rectangle (3,1);
```



```
\tikz \draw [top color=orange,shading angle=45] (0,0) rectangle (3,1);
```



```
\tikz \draw [bottom color=orange] (0,0) rectangle (3,1);
```



## La couleur d'un dégradé

Changer la couleur d'un dégradé est un peu compliqué et dépend du type de dégradé. On va étudier tout d'abord le dégradé axial. Il y a deux commandes pour le faire : `top color=couleur` et `bottom color=couleur`. Quand tu utilises ces options tu passes automatiquement en mode `shade` avec `shading=axis` et `shade angle=0`, donc pas besoin de les préciser. Tu peux toutefois changer l'angle. Par défaut `top color=gris` et `bottom color=blanc`. Exemples :

```
\tikz \draw [top color=orange] (0,0) rectangle (3,1);
```



```
\tikz \draw [top color=orange,shading angle=45] (0,0) rectangle (3,1);
```



```
\tikz \draw [bottom color=orange] (0,0) rectangle (3,1);
```



```
\tikz \draw [top color=blue,bottom color=orange] (0,0) rectangle (3,1);
```



Introduction

Le système de  
coordonnées

Les coordonnées  
relatives

Les coordonnées  
relatives

Les chemins : path

Les opérations

Les actions

La couleur

Position des options

Les actions (suite)

conclusion

## La couleur médiane d'un dégradé

`middle color=couleur` change la couleur du milieu Exemples :

## La couleur médiane d'un dégradé

`middle color=couleur` change la couleur du milieu Exemples :

```
\tikz \draw [middle color=red] (0,0) rectangle (3,1.5);
```



## La couleur médiane d'un dégradé

`middle color=couleur` change la couleur du milieu Exemples :

```
\tikz \draw [middle color=red] (0,0) rectangle (3,1.5);
```



```
\tikz \draw [middle color=red,shading angle=45] (0,0) rectangle (3,1.5);
```



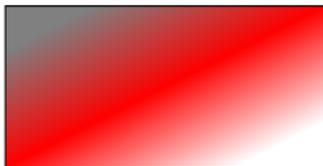
## La couleur médiane d'un dégradé

`middle color=couleur` change la couleur du milieu Exemples :

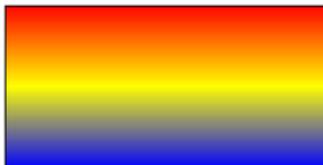
```
\tikz \draw [middle color=red] (0,0) rectangle (3,1.5);
```



```
\tikz \draw [middle color=red,shading angle=45] (0,0) rectangle (3,1.5);
```



```
\tikz \draw [bottom color=blue,top color=red,middle color=yellow] (0,0) rectangle (3,1.5);
```



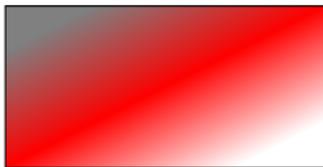
## La couleur médiane d'un dégradé

`middle color=couleur` change la couleur du milieu Exemples :

```
\tikz \draw [middle color=red] (0,0) rectangle (3,1.5);
```



```
\tikz \draw [middle color=red,shading angle=45] (0,0) rectangle (3,1.5);
```



```
\tikz \draw [bottom color=blue,top color=red,middle color=yellow] (0,0) rectangle (3,1.5);
```



## Attention

Si tu changes les couleurs du bas et du haut, la couleur du milieu doit être saisie à la fin.

## La couleur d'un dégradé

Les options `left color=couleur` et `right color=couleur` se comportent exactement comme `top color` et `bottom color` la seule différence est que `shading angle=90` et non pas 0. Exemples :

## La couleur d'un dégradé

Les options `left color=couleur` et `right color=couleur` se comportent exactement comme `top color` et `bottom color` la seule différence est que `shading angle=90` et non pas 0. Exemples :

```
\tikz \draw [left color=orange] (0,0) rectangle (3,1);
```



## La couleur d'un dégradé

Les options `left color=couleur` et `right color=couleur` se comportent exactement comme `top color` et `bottom color` la seule différence est que `shading angle=90` et non pas 0. Exemples :

```
\tikz \draw [left color=orange] (0,0) rectangle (3,1);
```



```
\tikz \draw [top color=orange,shading angle=45] (0,0) rectangle (3,1);
```



## La couleur d'un dégradé

Les options `left color=couleur` et `right color=couleur` se comportent exactement comme `top color` et `bottom color` la seule différence est que `shading angle=90` et non pas 0. Exemples :

```
\tikz \draw [left color=orange] (0,0) rectangle (3,1);
```



```
\tikz \draw [top color=orange,shading angle=45] (0,0) rectangle (3,1);
```



```
\tikz \draw [right color=orange] (0,0) rectangle (3,1);
```



## La couleur d'un dégradé

Les options `left color=couleur` et `right color=couleur` se comportent exactement comme `top color` et `bottom color` la seule différence est que `shading angle=90` et non pas 0. Exemples :

```
\tikz \draw [left color=orange] (0,0) rectangle (3,1);
```



```
\tikz \draw [top color=orange,shading angle=45] (0,0) rectangle (3,1);
```



```
\tikz \draw [right color=orange] (0,0) rectangle (3,1);
```



```
\tikz \draw [top color=blue,right color=orange] (0,0) rectangle (3,1);
```



## La couleur d'un dégradé

Pour changer les couleurs d'un dégradé radial, il faut utiliser `inner color=couleur` qui change la couleur du centre et `outer color=couleur` qui change la couleur extérieure . Tu passes, alors, automatiquement en mode `shade` avec `shading=radial`. Exemples :

## La couleur d'un dégradé

Pour changer les couleurs d'un dégradé radial, il faut utiliser `inner color=couleur` qui change la couleur du centre et `outer color=couleur` qui change la couleur extérieure . Tu passes, alors, automatiquement en mode `shade` avec `shading=radial`. Exemples :

```
\tikz \draw [inner color=orange] (0,0) rectangle (3,1);
```



## La couleur d'un dégradé

Pour changer les couleurs d'un dégradé radial, il faut utiliser `inner color=couleur` qui change la couleur du centre et `outer color=couleur` qui change la couleur extérieure . Tu passes, alors, automatiquement en mode `shade` avec `shading=radial`. Exemples :

```
\tikz \draw [inner color=orange] (0,0) rectangle (3,1);
```



```
\tikz \draw [outer color=orange] (0,0) rectangle (3,1);
```



## La couleur d'un dégradé

Pour changer les couleurs d'un dégradé radial, il faut utiliser `inner color=couleur` qui change la couleur du centre et `outer color=couleur` qui change la couleur extérieure. Tu passes, alors, automatiquement en mode `shade` avec `shading=radial`. Exemples :

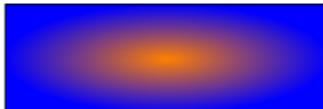
```
\tikz \draw [inner color=orange] (0,0) rectangle (3,1);
```



```
\tikz \draw [outer color=orange] (0,0) rectangle (3,1);
```



```
\tikz \draw [inner color=orange,outer color=blue] (0,0) rectangle (3,1);
```



Introduction

Le système de  
coordonnées

Les coordonnées  
relatives

Les coordonnées  
relatives

Les chemins : path

Les opérations

Les actions

La couleur

Position des options

Les actions (suite)

conclusion

## La couleur d'un dégradé

Pour changer la couleur des balles tu utilises `ball color=couleur`. Tu passe automatiquement en mode `shade` avec `shading=ball`. Tu ne peux pas changer la couleur de l'éclat de lumière qui reste blanc. Exemples :

## La couleur d'un dégradé

Pour changer la couleur des balles tu utilises `ball color=couleur`. Tu passe automatiquement en mode `shade` avec `shading=ball`. Tu ne peux pas changer la couleur de l'éclat de lumière qui reste blanc. Exemples :

```
\tikz \shade [ball color=orange] (0,0) circle (10pt);
```



## La couleur d'un dégradé

Pour changer la couleur des balles tu utilises `ball color=couleur`. Tu passe automatiquement en mode `shade` avec `shading=ball`. Tu ne peux pas changer la couleur de l'éclat de lumière qui reste blanc. Exemples :

```
\tikz \shade [ball color=orange] (0,0) circle (10pt);
```



```
\tikz \shade [ball color=black] (0,0) circle (10pt);
```



## La couleur d'un dégradé

Pour changer la couleur des balles tu utilises `ball color=couleur`. Tu passe automatiquement en mode `shade` avec `shading=ball`. Tu ne peux pas changer la couleur de l'éclat de lumière qui reste blanc. Exemples :

```
\tikz \shade [ball color=orange] (0,0) circle (10pt);
```



```
\tikz \shade [ball color=black] (0,0) circle (10pt);
```



```
\tikz \shade [ball color=red] (0,0) circle (10pt);
```



## Conclusion

On est loin d'avoir fait le tour de TikZ, mais tu as déjà de nombreux éléments pour commencer à dessiner des petits Mickey. On verra dans d'autres fiches comment placer tes dessins dans la page L<sup>A</sup>T<sub>E</sub>X, comment modifier les traits, utiliser les [node](#)...

On verra comment réaliser l'accordéoniste ci-dessous. Si si c'est un fichier TikZ et il a été généré par [pdflatex](#), mais il y a un truc.

