

se positionner sur le
canvas

Tracer une courbe ?

un boucle sur “t”

les abscisses et les
ordonnées

changer de référentiel

L'accroissement en x

L'outil CANVAS

User Interface

Layout

Media

Drawing and Animation

Ball

Canvas

ImageSprite

Maps

Charts

Chart

ChartData2D

Trendline

Data Science

Sensors

Social

Storage

Connectivity

LEGO® MINDSTORMS®

Experimental

Extensions

Canvas

A two-dimensional touch-sensitive rectangular panel on which drawing can be done and sprites can be moved.

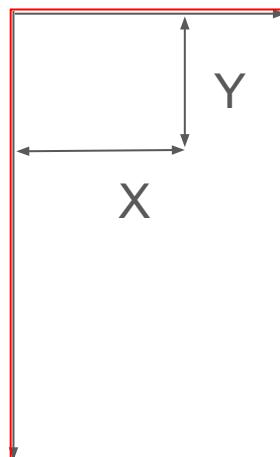
The `BackgroundColor`, `PaintColor`, `BackgroundImage`, `Width`, and `Height` of the Canvas can be set in either the Designer or in the Blocks Editor. The `Width` and `Height` are measured in pixels and must be positive.

Any location on the Canvas can be specified as a pair of (X, Y) values, where

- X is the number of pixels away from the left edge of the Canvas
- Y is the number of pixels away from the top edge of the Canvas

There are events to tell when and where a Canvas has been touched or a Sprite (ImageSprite or Ball) has been dragged. There are also methods for drawing points, lines, and circles.

[More information](#)



Screen1

MENU_HorizontalArran

MENU_Button1

Canvas2

CANVAS_VerticalArran

Canvas1

gommeSprite

Rename Delete

Media

eraser-6..._720.webp

Upload File ...

call `Canvas1 .DrawLine`
`x1` `y1` `x2` `y2`

call `Canvas1 .DrawPoint`
`x` `y`

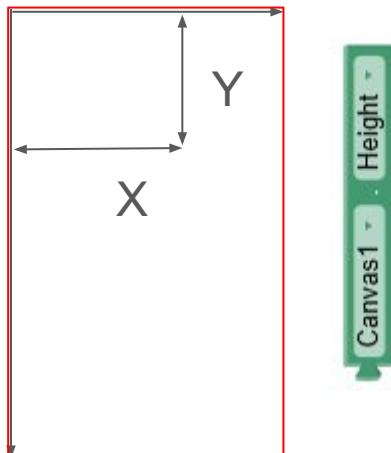
call `Canvas1 .DrawShape`
`pointList` `fill` `true`

call `Canvas1 .DrawText`
`text` `x` `y`

2

De Canvas aux fonctions mathématiques

Canvas1 . Width

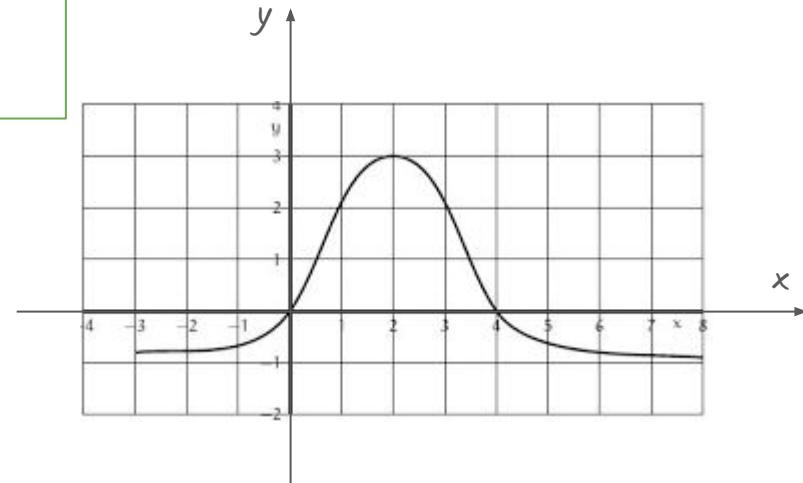


si $x = -4$ alors $X = 0$

si $x = 8$ alors $X = \text{Canvas.width}$

si $y = -2$ alors $Y = \text{Canvas.height}$

si $y = 4$ alors $Y = 0$

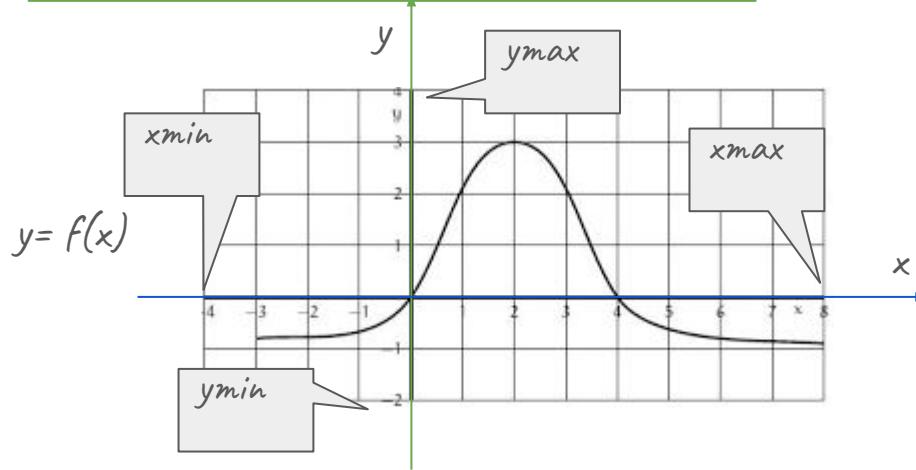


$$y = f(x)$$

Blocs de changement de coordonnées

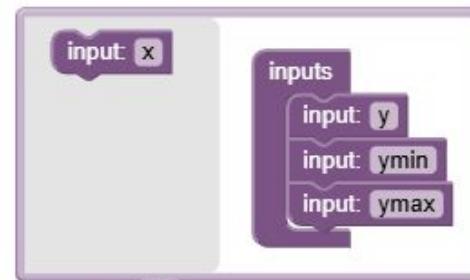
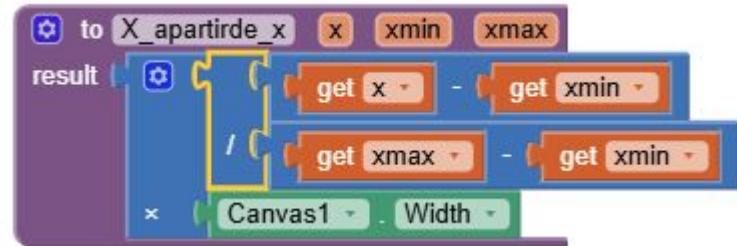
si $x = x_{min}$ alors $X = 0$

si $x = x_{max}$ alors $X = \text{Canvas.width}$



si $y = y_{min}$ alors $Y = \text{Canvas.height}$

si $y = y_{max}$ alors $Y = 0$

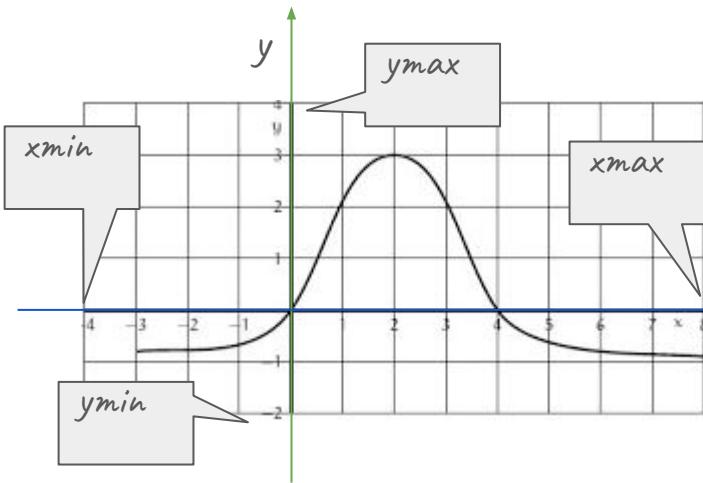


Traçage des axes

```
call [Canvas1 v].DrawLine  
x1  
y1  
x2  
y2
```



```
call [Canvas1 v].DrawPoint  
x  
y
```



```
call [Canvas1 v].DrawLine  
x1  
y1  
x2  
y2
```

```
call [Canvas1 v].DrawCircle  
centerX  
centerY  
radius  
fill
```

```
for each [number] from [floor y] to [ceiling y] by [1]  
do  
    call [Canvas1 v].DrawLine  
        x1  
        y1  
        x2  
        y2  
    call [Canvas1 v].DrawCircle  
        centerX  
        centerY  
        radius  
        fill
```

5

Une boucle pour tracer la fonction

sin

✓ sin

cos

tan

asin

acos

atan

square root

✓ square root

absolute

neg

log

e^A

round

ceiling

floor

```
to courbe1 [xmin xmax ymin ymax radius]
do [initialize local height to (Canvas1).Height]
in [initialize local width to (Canvas1).Width]
in [set (Canvas1).PaintColor to black]
call LOG_message [blockName...]
for each number from (get xmin) to (get xmax) by 0.1
do [initialize local X to call X_apartirde_x [x (get number) xmin (get xmin) xmax (get xmax) canvasWidth (get width)]]
in [initialize local Y to call Y_apartirde_y [y (2 * (sin (get number) * (180 / 3.14))) ymin (get ymin) ymax (get ymax) canvasHeight (get height)]]
call Canvas.DrawCircle [for component (Canvas1) centerX (get X) centerY (get Y) radius (get radius) fill true]
call LOG_message [blockName...]
```

