The canvas package

Johannes Wolf and fenjalien

https://github.com/johannes-wolf/typst-canvas

https://github.com/fenjalien/typst-canvas

Contents

1. Introduction

This package provides a way to draw stuff using a similar API to [Processing](#) but with relative coordinates and anchors from [Tikz](#). You also won't have to worry about accidentally drawing over other content as the canvas will automatically resize. And remember up is negative!

2. Usage

This is the minimal starting point:

```
#import "typst-canvas/canvas.typ": canvas

#canvas({
  import "typst-canvas/draw.typ": *
  ...
})
```

Note that draw functions are imported inside the scope of the `canvas` block. This is recommended as draw functions override Typst's functions such as `line`.

2.1. Coordinates

There are four different ways to specify coordinates.

1. Absolute: `(x,y)`
   "x units to the right and y units down from the origin."
2. Relative: `(rel: (x,y))`
   "x units to the right and y units down from the previous coordinate."
3. Previous: `()`
   "The previous coordinate."
4. Anchor: `(node: "name", at: "example")` or `"name.example"`
   "The position of anchor `"example"` on node with name `"name"`."
   See Section 2.2

2.2. Anchors

Anchors are named positions relative to named elements.

To use an anchor of an element, you must give the element a name using the `name` parameter.
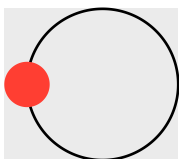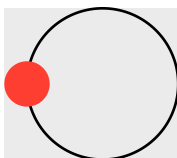
```
#canvas({
  import "typst-canvas/draw.typ": *
  // Name the circle
  circle((0,0), name: "circle")

  // Draw a smaller red circle at "circle"'s left anchor
  fill(red)
  stroke(none)
  circle("circle.left", radius: 0.3)
})
```

All elements will have default anchors based on its bounding box, they are: `center`, `left`, `right`, `above` and `below`. Some elements will have their own anchors.

Elements can be placed relative to its own anchors.

```
#canvas({
  import "typst-canvas/draw.typ": *
  // An element does not have to be named
  // in order to use its own anchors.
  circle((0,0), anchor: "left")

  // Draw a smaller red circle at the origin
  fill(red)
  stroke(none)
  circle("circle.left", radius: 0.3)
})
```

3. Reference

```
#canvas(background: none, length: 1cm, debug: false, body)
```

background A color to be used for the background of the canvas.

length The length used to specify what 1 coordinate unit is.

debug Shows the bounding boxes of each element when true.

body A code block in which functions from draw.typ have been called.

3.1. Elements

```
#line(start, end, mark-begin: none, mark-end: none, name: none)
```

Draws a line (a direct path between two points) to the canvas.

start The coordinate to start drawing the line from

end The coordinate to draw the line to.

mark-begin The type of arrow to draw at the start of the line.

mark-end The type of arrow to draw at the end of the line.
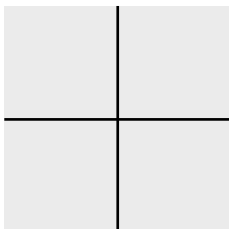
```
#canvas({
  import "typst-canvas/draw.typ": *
  line((-1.5, 0), (rel: (3, 0)))
  line((0, -1.5), (rel: (0, 3)))
})
```

```
#rect(a, b, name: none)
```

Draws a rectangle to the canvas.

a The top left coordinate of the rectangle.

b The bottom right coordinate of the rectangle.

```
#canvas({
  import "typst-canvas/draw.typ": *
  rect((-1.5, 1.5), (1.5, -1.5))
})
```

```
#arc(position, start, stop, radius: 1, name: none, anchor: none)
```

Draws an arc to the canvas.

position The coordinate to start drawing the arc from.

start The angle to start the arc.

stop The angle to stop the arc.

radius The radius of the arc's circle.

```
#canvas({
  import "typst-canvas/draw.typ": *
  arc((0,0), 45deg, 135deg)
})
```

```
#circle(center, radius: 1, name: none, anchor: none)
```

Draws a circle to the canvas.

center The coordinate of the circle's origin.

radius The circle's radius.

```
#canvas({
  import "typst-canvas/draw.typ": *
  circle((0,0))
})
```

```
#content(pt, ct, angle: 0deg, name: none, anchor: none)
```

Draws a content block to the canvas.

pt The coordinate of the center of the content block.

ct The content block.

angle The angle to rotate the content block by. Uses Typst's `rotate` function.

```
#canvas({
  import "typst-canvas/draw.typ": *
  content((0,0), [Hello World!])
})
```

3.2. Styles