P.PORTO

POLITÉCNICO
DO PORTO
ESMAD

COMPUTAÇÃO GRÁFICA
TSIW

Syllabus

- HTML element: Canvas
- Context 2D and coordinates
- Simplest shape: rectangle

Why Canvas?

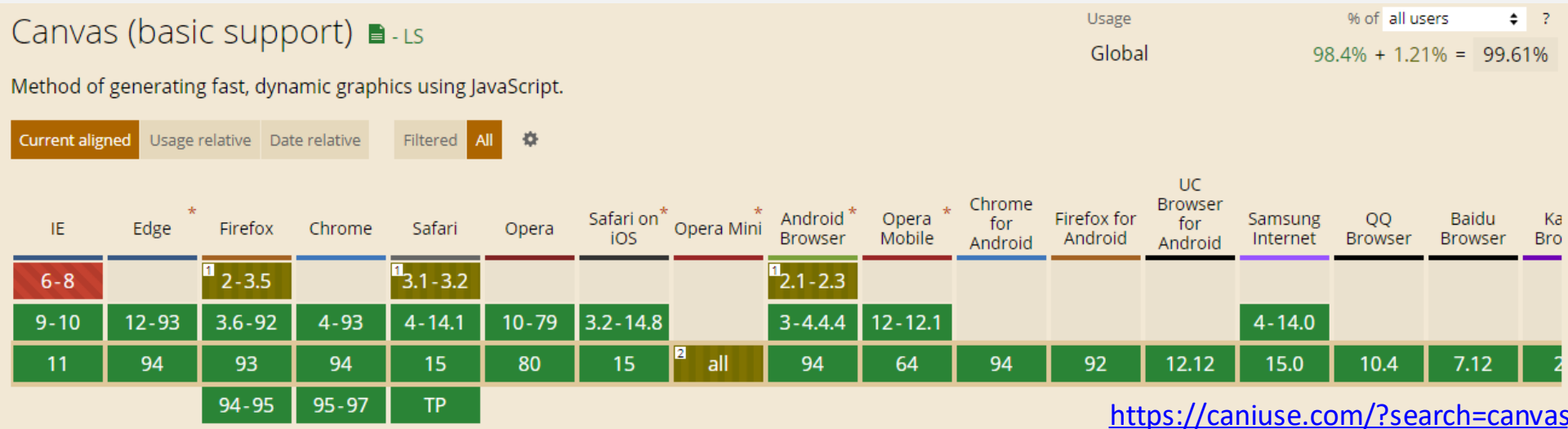
- Interactivity
- Animation
- Flexibility
- Browser/Platform Support
- Popularity
- Web standard
- Develop once, run everywhere
- Free and accessible development tools

HTML element: Canvas

- HTML5 specs includes new functionalities, one of them being the Canvas element

"resolution-dependent bitmap canvas, which can be used for rendering graphs, game graphics, art, or other visual images on the fly"

- Drawing and animation is performed by scripting (**JS**)



HTML element: Canvas

HTML

```
<!DOCTYPE html>
<html>
   <head>
       <title>Your First Canvas Application </title>
   </head>

   <body>
        <canvas>
            Your browser does not support HTML5 Canvas.
        </canvas>
   </body>
</html>
```

What do you see on your browser?

HTML element: Canvas

HTML

```
<!DOCTYPE html>
<html>
    <head>
        <title>Your First Canvas Application </title>
         <style>
             canvas {
                 border: 3px solid orange;
             }
         </style>
    </head>

    <body>
         <canvas>
             Your browser does not support HTML5 Canvas.
         </canvas>
    </body>
</html>
```

CSS

And now?

HTML element: Canvas

- Canvas element has no content nor border
- There can be more than one Canvas element in a single webpage
- Default size is `300px` by `150px`
- Like other HTML elements, it has several attributes:
  `id`, `width`, `height`, `style`, …

```
<canvas id="canvas1" width="200" height="100" style="border:1px solid blue;">
    Your browser does not support HTML5 Canvas.
</canvas>
```

- Text between tags is the *fallback content*
- Given an `id`, it is **accessible using JavaScript**

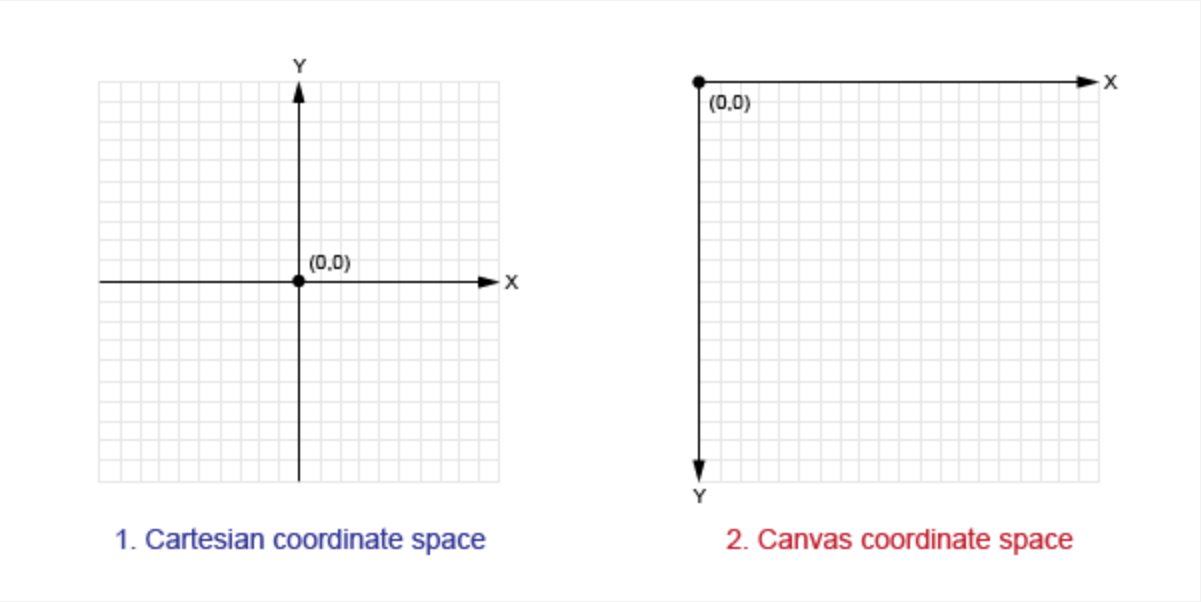
JS object: HTMLCanvasElement

JS

```
// Getting the DOM element
const canvas = document.querySelector("#canvas1");

// Getting its 2D rendering context
const ctx = canvas.getContext("2d");
```

- **`canvas`**: Canvas object in JavaScript
- Two properties: `width` and `height`
  - sets both the element's size and the size of the element's drawing surface
- Three methods, being the principal `getContext()`:
  - sets the **rendering context** bound to it: 2D, webgl, webgl2 (),…

Canvas grid coordinates

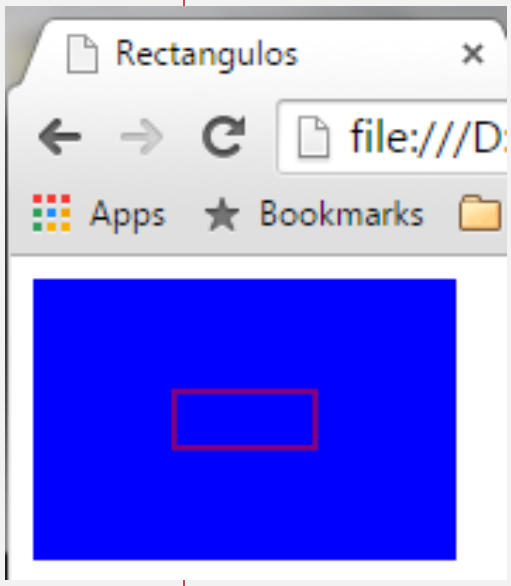


- Points outside the Canvas grid boundaries (`width` and `height`) are not drawn
- It is possible to modify the Canvas coordinate system using transformations (will be learned later…)

Simplest shape: rectangle

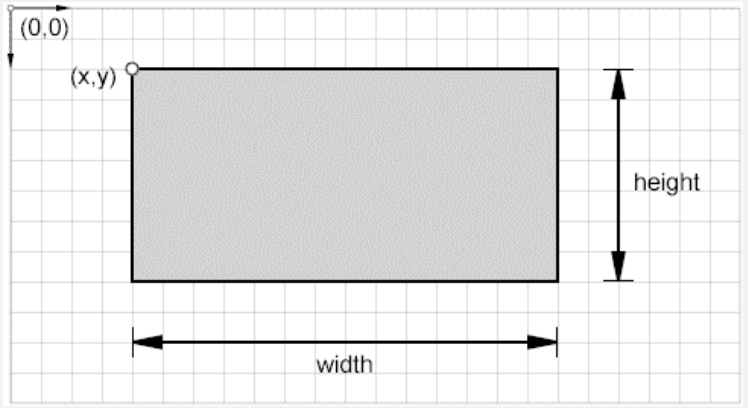
JS

```
// Getting the DOM element
const canvas = document.querySelector("#canvas1");

// Getting its 2D rendering context
const ctx = canvas.getContext("2d");

// Drawing a blue solid rectangle
ctx.fillStyle = 'blue';
ctx.fillRect(0, 0, 150, 100);

// Drawing a smaller red rectangle
ctx.strokeStyle = 'red';
ctx.strokeRect(50, 40, 50, 20);
```



- All Canvas drawings must be performed using JS **in its rendering context** - in the example: **ctx**

Simplest shape: rectangle



`fillRect(x,y,width,height)`

paints the given rectangle, using the current fill style

`strokeRect(x,y,width,height)`

paints the box that outlines the given rectangle, using the current stroke style

`clearRect(x,y,width,height)`

clears all pixels in the given rectangle to transparent black

`rect(x,y,width,height)`

draws the given rectangle has part of a path (will be learned later…)

Try yourself…

1. Can you try to make flags of countries like:
   Germany
   Russia
   Sweden
   Switzerland
   France

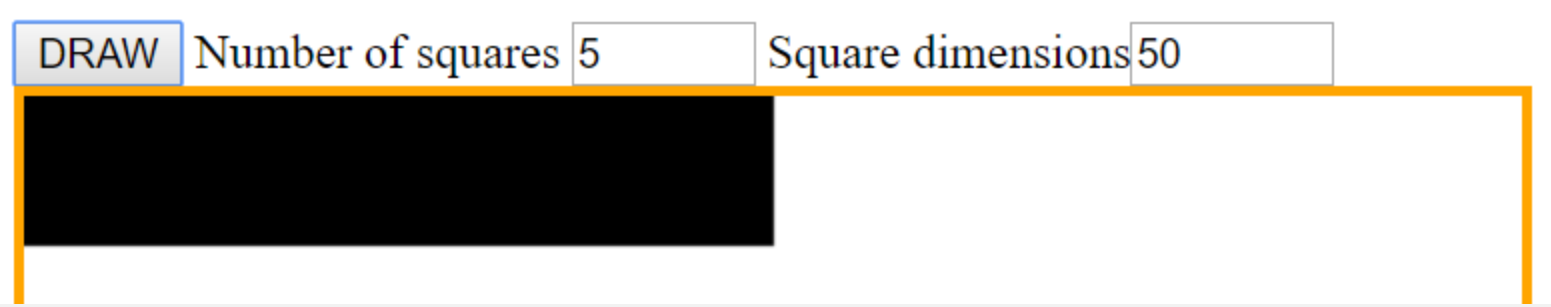
- Download from Moodle file `Ex01_flags.html` and code the solution in where you find the comments `//TODO`

Try yourself…

2. Now make use of your algorithmic skills to draw a configurable black-and-white checkerboard

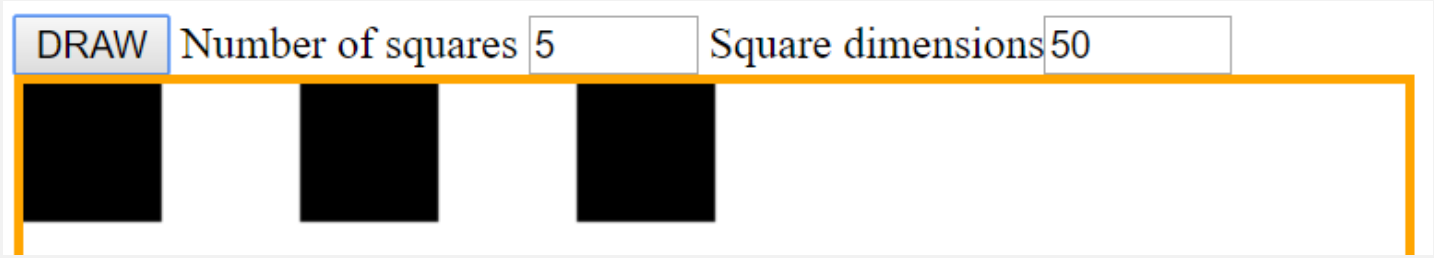
a) Download from Moodle file `Ex02_checkerboard.html` and code the solution in where you find the comment `//TODO`

b) Draw 1 row of N black squares of size D

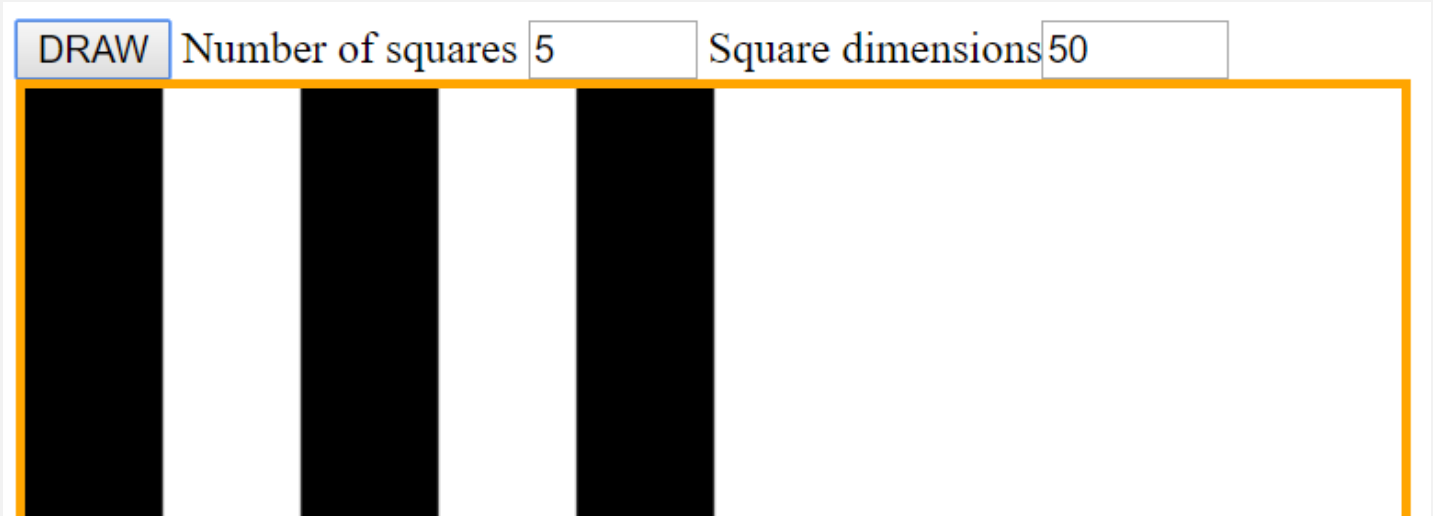


Try yourself…

c) From that row, skip drawing the pair squares

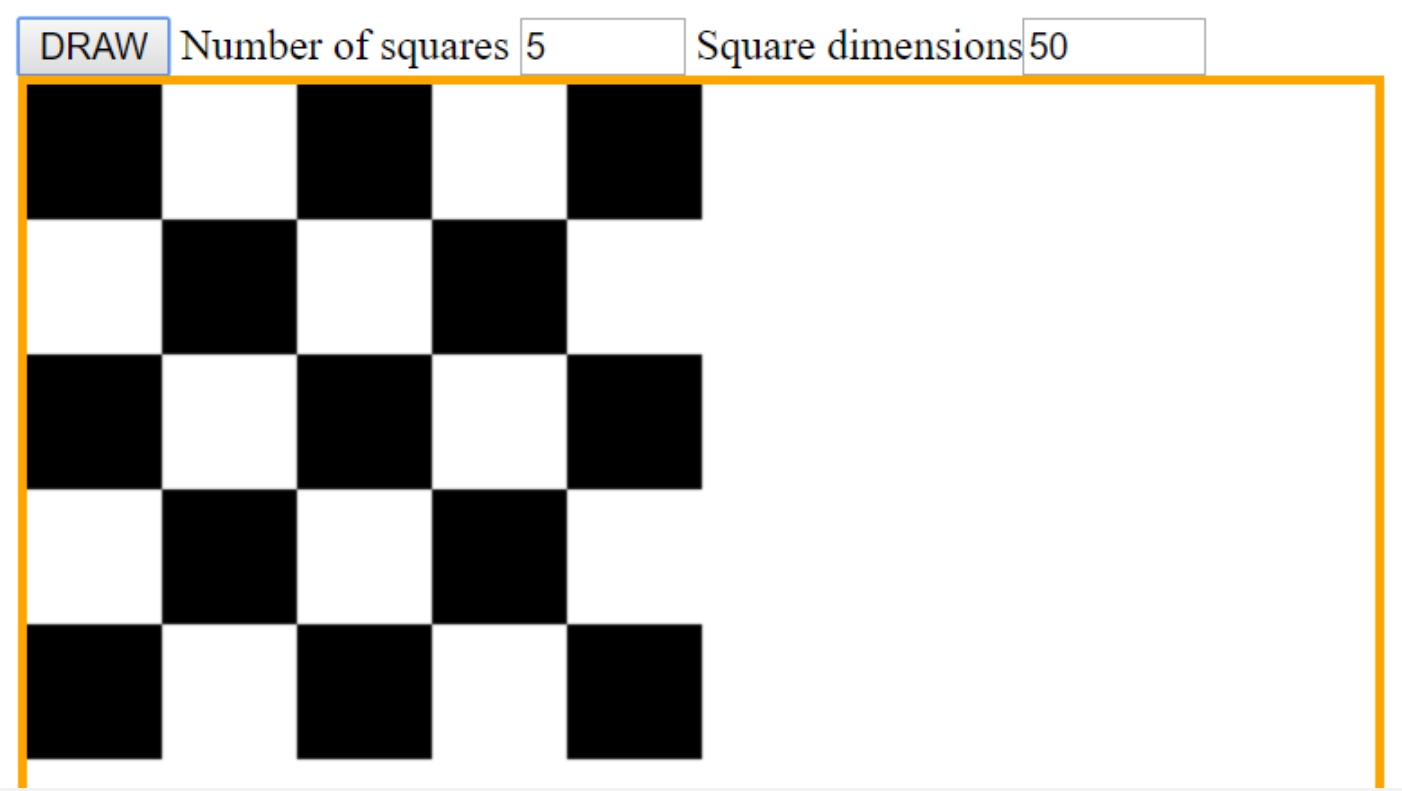


d) Draw N rows of N black squares, skipping the pair squares



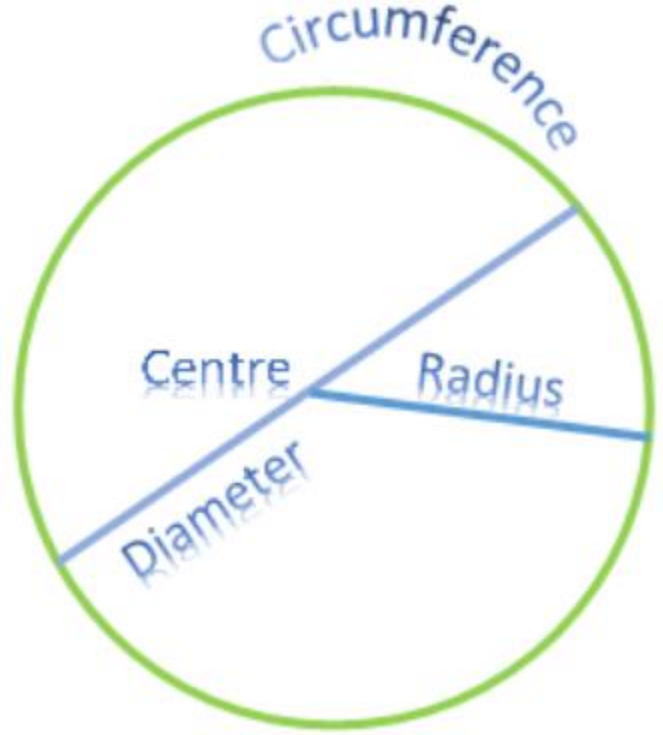
Try yourself…

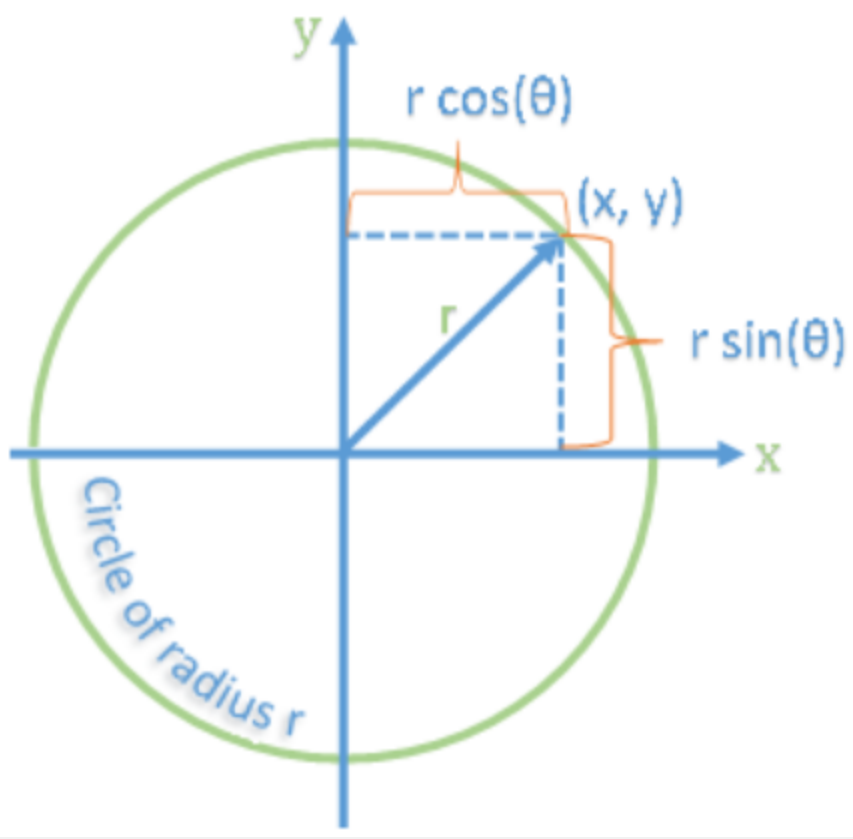
e) Now, fix the skipping part of the algorithm to draw a proper checkerboard



Try yourself…

3. Finish by refreshing some 2D geometry





Try yourself…

a) Download from Moodle file `Ex03_boxesInCircle.html` and code the solution in where you find the comment `//TODO`

b) Knowing the:

– **Parametric** circumference equations:

$$\begin{cases} x = x_0 + r\cos t \\ y = y_0 + r\sin t \end{cases}$$

– $360° = 2\pi$ radians

Draw N circles equally spaced around a circle, centered in the Canvas element, with a radius of 100 pixels.

Try yourself…