Responsive canvas, resizing a canvas

INTRODUCTION

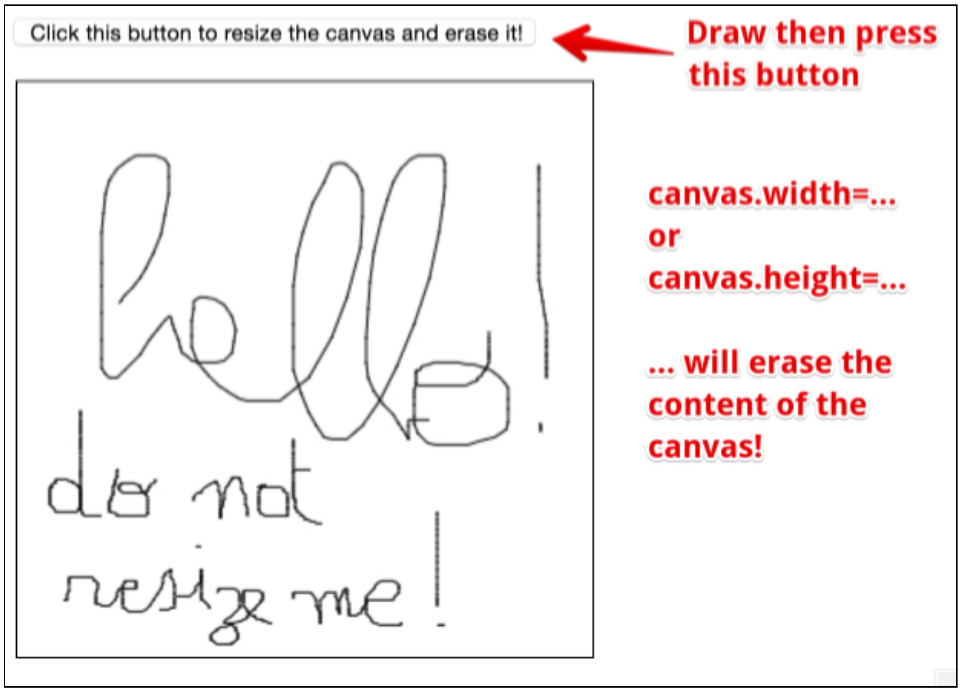
Resizing a canvas can be tricky if we don't know a few rules that might not be easily guessed:

1. Changing the `width` or `height` property of a canvas in JavaScript erases its content and resets its context,
2. Using percentages (%) in the CSS `width` and `height` properties of a canvas does not change its number of pixels/resolution. Instead, it scales the existing pixels without erasing the content, giving a blurry effect when a canvas becomes larger for example.

Before looking at how best to handle canvas resizing, let's see some examples below:

EXAMPLE 1: CHANGING THE SIZE OF A CANVAS ON THE FLY ERASES ITS CONTENT!

Online example: http://jsbin.com/tukave/2/edit



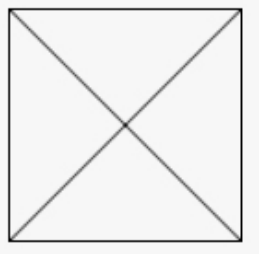

```
<script>
...
 function resizeCanvas() {
    canvas.width = 300;
 }

</script>
...
<button onclick="resizeCanvas();">
    Click this button to resize the canvas and erase it!
</button>
```

EXAMPLE 2 : RESIZE A CANVAS USING CSS WIDTH AND HEIGHTPROPERTIES WITH PERCENTAGES

This time we are using a similar example as above, but we removed the button for resizing it, and we set the size of the canvas to 100x100 pixels. Instead of drawing inside, we draw two lines that join the diagonals.

Here is the online version : http://jsbin.com/luvuhe/4/edit

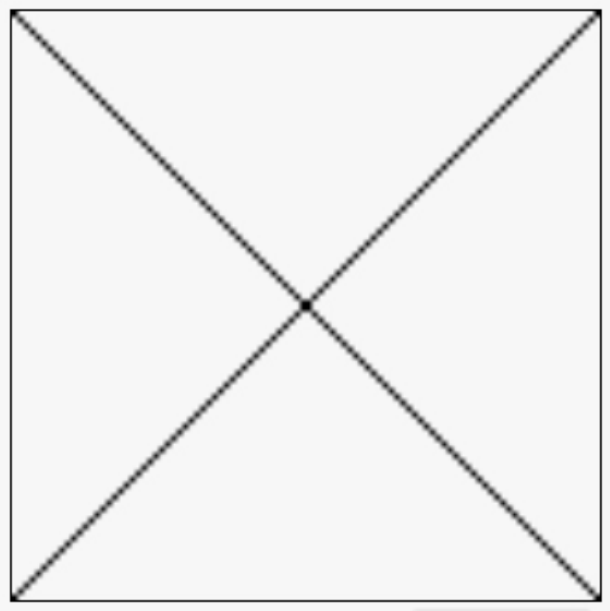


Then, we added this CSS rule, try it online (resize the windows, you will see what happens): http://jsbin.com/pajifo/2/edit

It's the same example as before, just adding the CSS:

```
<style>
  #myCanvas {
    border: 1px solid black;
    width:100%
  }
</style>
```

And the result shows clearly that the resolution is still the same, only the pixels are bigger!



Even bigger:

GOOD PRACTICE:* *never use CSS percentages on a canvas width or height!

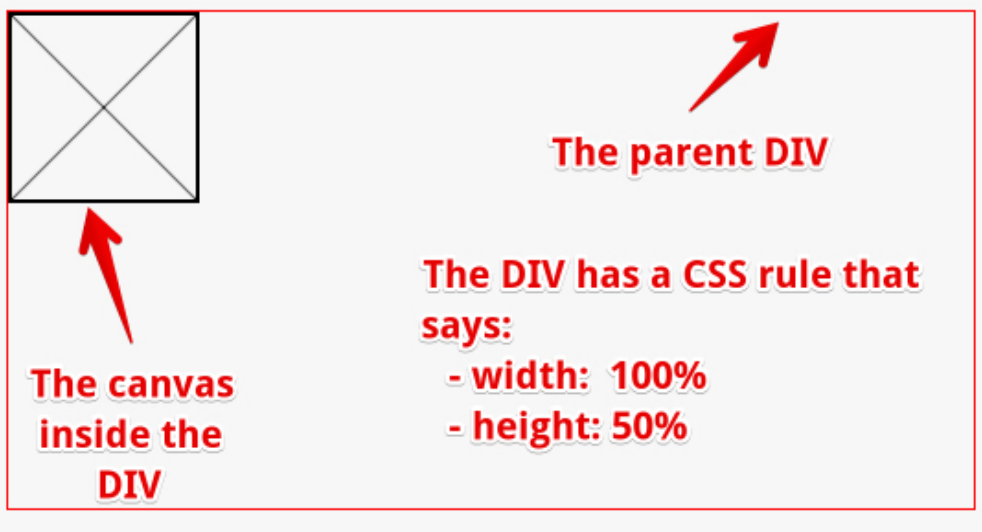
EXAMPLE 3: A RESPONSIVE CANVAS USING A RESIZE LISTENER + A PARENT ELEMENT

This is the trick in order to have a really responsive canvas:

1. embed it in a `<div>` or in any parent container,
2. Use CSS with percentages on the `width` and the `height` CSS properties **of the parent,**
3. Use a `resize` listener on the parent of the canvas,
4. Change the `width` and `height` properties of the canvas from the JavaScript resize listener function (content will be erased),
5. Redraw the content, scaled accordingly to the size of the parent.

Yep, this is not a straightforward process...

HTML, CSS and JavaScript code



HTML code:

```
<div id="parentDiv">
  <canvas id="myCanvas" width="100" height="100" ></canvas>
</div>
```

CSS code:

```
<style>
  #parentDiv {
     width:100%;
     height:50%;
     margin-right: 10px;
     border: 1px solid red;
  }

  canvas {
     border: 2px solid black;
  }
</style>
```

JavaScript code for the `resize` event listener:

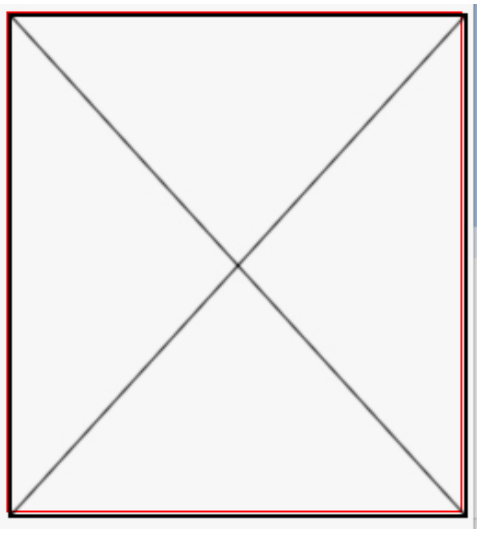
```
function init() {
  ...
  // IMPORTANT: there is NO WAY to listen to a DIV's resize
```

```
  // listen to the window instead.
  window.addEventListener('resize',
                          resizeCanvasAccordingToParentSize, false);
  ...
}

function resizeCanvasAccordingToParentSize() {
   // adjust canvas size, take parent's size, this erases content
   canvas.width = divcanvas.clientWidth;
   canvas.height = divcanvas.clientHeight;
   ...
   // draw something, taking into account the new canvas size
}
```

Complete example: http://jsbin.com/tubufo/2/edit

Original window size:

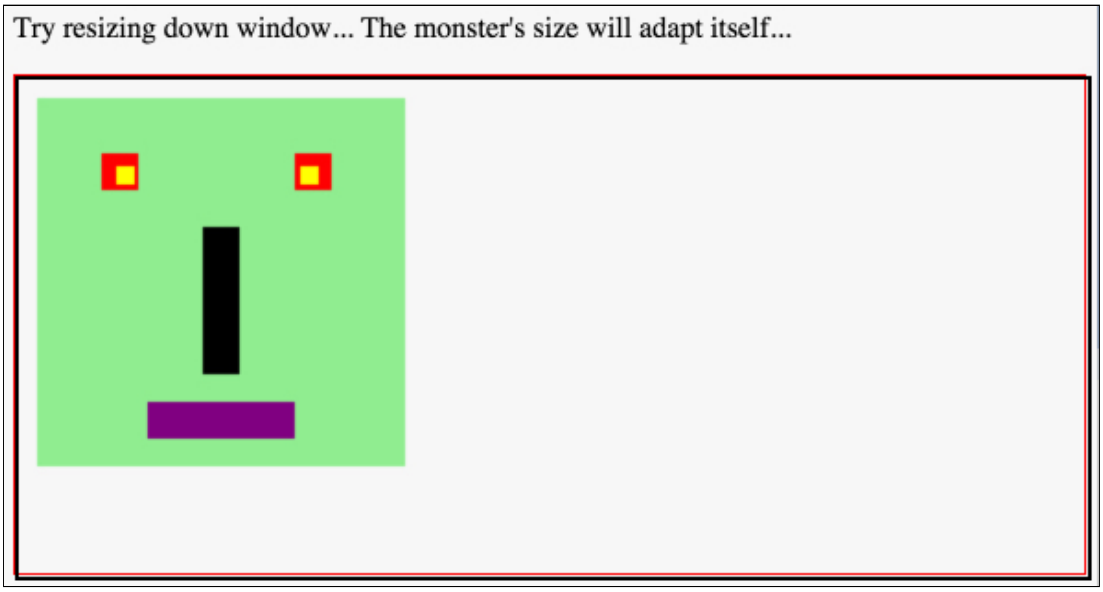


We resize the window horizontally:

EXAMPLE 4: THE SAME EXAMPLE WITH THE MONSTER

Online example: http://jsbin.com/xoqipo/3/edit

Initial size:



Canvas resized, width became smaller than the monster's size. We **scaled** down the monster (using `ctx.scale`!)

The code is very similar to the previous example, we just replaced `drawDiagonals()` by`drawMonster(...)`, and we added a test in the `drawMonster(...)` function for scaling the monster if it's bigger than the canvas width (look at lines 10-16), this is a common trick:

```
function drawMonster(x, y, angle, headColor, eyeColor) {
    // GOOD PRACTICE : SAVE CONTEXT AND RESTORE IT AT THE END
    ctx.save();
    // Moves the coordinate system so that the monster is drawn
    // at position (x, y)
    ctx.translate(x, y);
    ctx.rotate(angle);
10. // Adjust the scale of the monster (200x200) if the canvas
11. // is too small
    if(canvas.width < 200) {
        var scaleX = canvas.width/200;
        var scaleY = scaleX;
    }
    ctx.scale(scaleX, scaleY);
    // head
    ctx.fillStyle=headColor;
    ctx.fillRect(0,0,200,200);
    ...
}
```

KNOWLEDGE CHECK 4.3.4 (NOT GRADED)

```
#myCanvas {
       border: 1px solid black;
       width:100%
}
```

Using CSS % for resizing a canvas is?

- Ok
- Bad practice
- Recommended
- Not possible