Drawing principles

MORE ABOUT THE "CONTEXT" OBJECT

Before we go on, we should take some time to clarify the way we draw on HTML5 canvases. We already mentioned that we use a graphic context for all the main operations. Whenever a shape, a text, or an image is drawn, the current values of the different properties of the graphic context are taken into account. Some are relevant only for certain kinds of shapes or drawing modes, but you must be aware that it is always the current values of these drawing properties that are used.

Later on we'll see that there are ways to save and restore this whole set of values, but for now, let's examine in greater detail some of the properties and methods we've already encountered, and introduce new ones.

MORE ABOUT PROPERTIES AND METHODS OF THE CONTEXT OBJECT

`fillStyle` is a property of the context, similar in a way to a CSS property.

Its value can be one of the following:

- a color,
- a pattern (texture), or
- a gradient.

The default value is the color black. Any kind of drawing in "fill mode" will use the value of this property to determine how to render the "filled part" of the drawing: any filled rectangle will be filled black by default, any filled circle will be filled in black, and so on.

As long as we don't modify the value of this property, all drawing commands for filled shapes will use the current value.

Note that we will study in detail how to use colors, gradients and patterns later, but for now we introduce some properties and values so that you can understand the principles of canvas drawing.

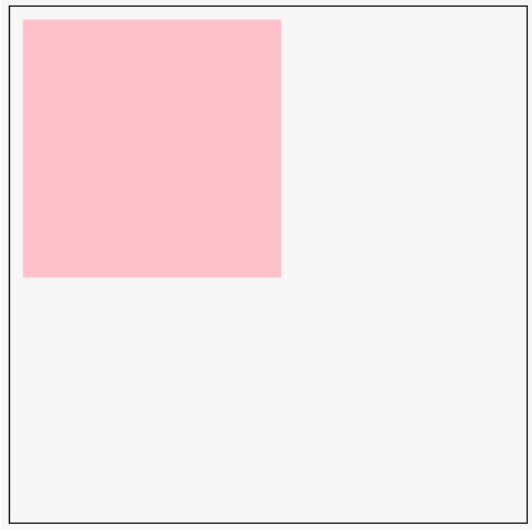
`fillStyle` and the other context properties can be considered to be "global variables" of the context.

`fillRect(x, y, width, height)`: a call to this method draws a filled rectangle.

The two first parameters are the coordinates of the top left corner of the rectangle. This method uses the current value of the `filledStyle` property to determine how to fill the rectangle.

```
ctx.fillStyle='pink';
ctx.fillRect(10,10,200,200);
```

Produces this result:



`strokeStyle` is a property of the context similar to `fillStyle`, but this time for indicating how the shape's outline should be rendered.

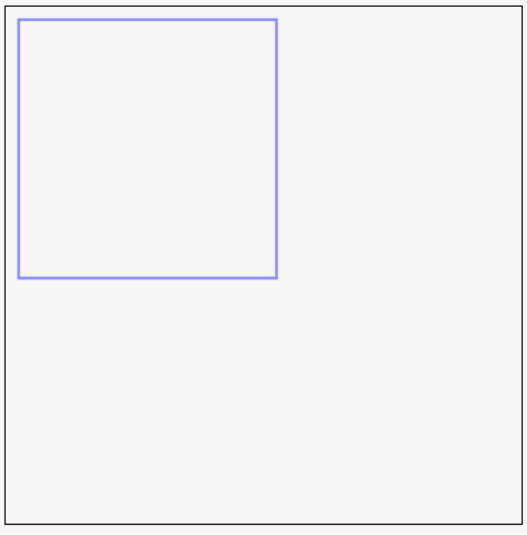
The possible values are the same as those for the `fillStyle`property: a color, a pattern, or a gradient. This property will be taken into account when wireframe shapes are

drawn.

`strokeRect(x, y, width, height)`: like `fillRect(...)`, but instead of drawing a filled rectangle the rectangle is drawn in wireframe mode.

```
ctx.strokeStyle='blue';
ctx.strokeRect(10,10,200,200);
```

...produces this result:



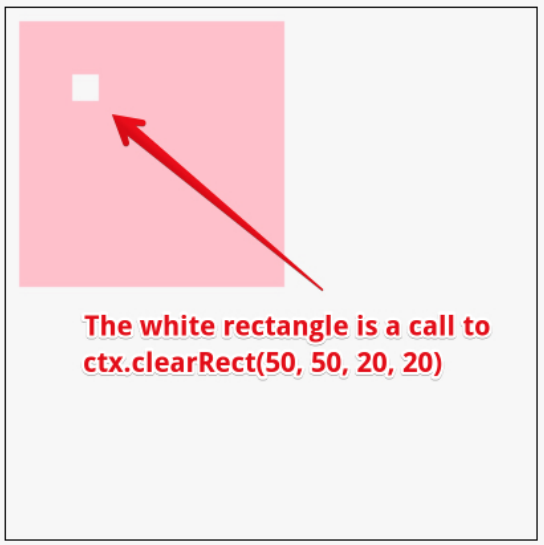
Only the outline of the rectangle will be drawn, and it will be drawn using the value of the `strokeStyle` property.

`clearRect(x, y, width, height)`: a call to this method erases the specified rectangle.

Actually it draws it in a color called "transparent black" (!) that corresponds to the initial state of the rectangle as if no drawing had occurred.

```
ctx.fillStyle='pink';
ctx.fillRect(10,10,200,200);
ctx.clearRect(50, 50, 20, 20);
```

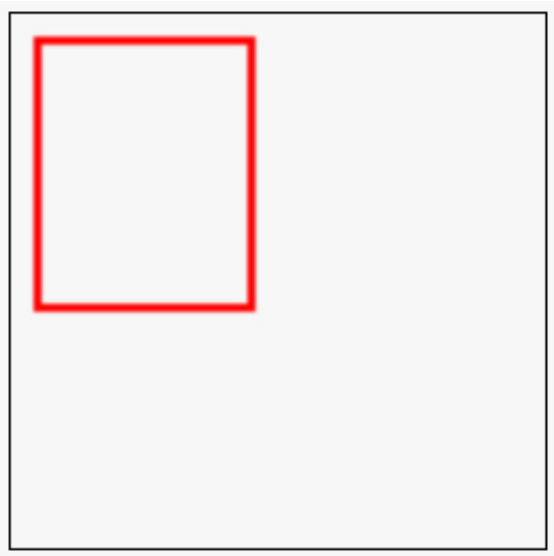
The result is:



LET'S SEE SOME SIMPLE EXAMPLES...

Draw a wireframe red rectangle, width `lineWidth`= 3 pixels.

Interactive example available here at JS Bin



Extract from the source code (the part that draws the rectangle):

```
function drawSomething() {
    // draw a red rectangle, line width=3 pixels
    ctx.lineWidth=3;
```

```
    ctx.strokeStyle='red';
    ctx.strokeRect(10,10,80,100);
}
```

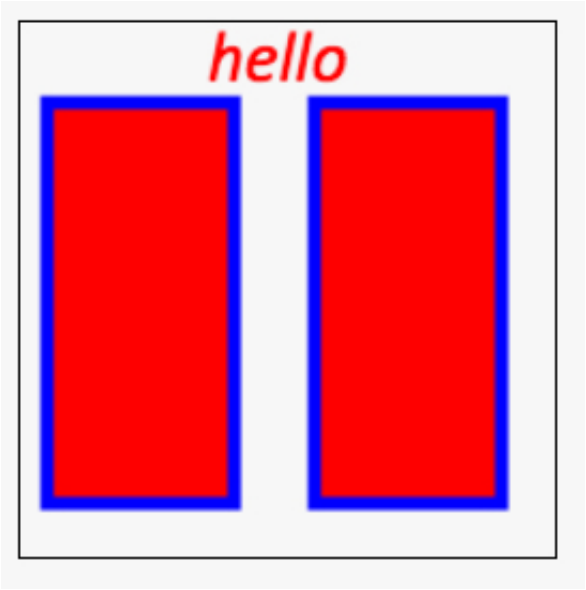
Here, we used "stroke" instead of "fill" in the property and method names (lines 4 and 5): `strokeStyle` instead of`fillStyle`, `strokeRect`(...) instead of `fillRect`(...).

We also introduced a new property of the context, that applies only when drawing in "stroke" mode, the `lineWidth` property (line 3), that is used for setting the width of the shape outline. The value is in pixels.

DRAW 2 FILLED RED RECTANGLES WITH A BLUE OUTLINE OF 5 PIXELS AND SOME TEXT

Let's continue with another example. This time we will draw several shapes that share the same colors - they will be filled in red, with a blue outline. We also show how to draw a text message with a given font.

Online example on JS Bin



Source code extract:

```
    function drawSomething() {
```

```
     // set the global context values
    ctx.lineWidth=5;
    ctx.fillStyle='red';
    ctx.strokeStyle='blue'
    // font for all text drawing
    ctx.font = 'italic 20pt Calibri';
    // Draw the two filled red rectangles
10. ctx.fillRect(10, 30, 70, 150);
    ctx.fillRect(110, 30, 70, 150);
    // Draw the two blue wireframe rectangles
    ctx.strokeRect(10, 30, 70, 150);
    ctx.strokeRect(110, 30, 70, 150);
    // Draw a message above the rectangles
    ctx.fillText("hello", 70, 22);
  }
```

This example shows the "global" nature of the context properties. Once you set the filled color to red, any shapes you draw in filled mode will be red. This is true for all the context properties. We set some of these properties in lines 3-7, and all following calls to context methods for drawing rectangles or text will depend on them. The two filled rectangles at lines 10-11 will be red, the two wireframe rectangles drawn at lines 14-15 will be blue, etc.

Line 18 shows how to draw a text message at an X position of 70 and a Y position of 22. The font is set at line 7 using the `font`property of the context. The syntax is the same we use in CSS for using "system fonts".

If you would like to draw the filled text message in green, for example, you should set the `ctx.fillStyle` property to "green" after you draw the rectangles and before you draw the text (i.e just before line 18).

SUMMARY OF WHAT WE LEARNED

- "stroke" means "wireframe" or "outlined", it is a prefix for setting properties or calling methods that will affect wireframe shapes, "fill" is a prefix for filled shapes.
- To set the properties of wireframe shapes use`ctx.`**`stroke`**`Style= ...,` for filled

shapes use `ctx.fillStyle=...` So far the values we have used are colors, expressed as strings. Example: `ctx.strokeStyle = 'blue';`

- To draw a wireframe rectangle use `ctx.strokeRect(x, y, width, height)`, to draw a filled rectangle use `ctx.fillRect(x, y, width, height);`
- To set the line width of wireframe shapes, use the `ctx.lineWidth` property. Example `ctx.lineWidth = 10;ctx.strokeRect(0, 0, 100, 100);` will draw a 100x100 rectangle in wireframe mode, with an outline width of 10 pixels.
- To draw a text message use `ctx.strokeText(message, x, y)` or `ctx.fillText(message, x, y)`, for wireframe text or filled text respectively.
- To set the character font use the `ctx.font` property; the value is a font in CSS syntax, for example: `ctx.font = 'italic 20pt Calibri';`

KNOWLEDGE CHECK 3.2.7 (NOT GRADED)

How would you draw a blue wireframe rectangle at 10, 10, width=200, height=400 with a lineWidth of 10 pixels?

- ctx.line=10; ctx.color='blue'; ctx.strokeRect(10, 10, 200, 400);
- ctx.lineWidth=10; ctx.color='blue'; ctx.strokeRect(200, 400, 10, 10);
- ctx.lineWidth=10; ctx.strokeStyle='blue'; ctx.strokeRect(10, 10, 200, 400);
- ctx.width=10; ctx.color='blue'; ctx.strokeRect(10, 10, 200, 400);
