Drawing shadows

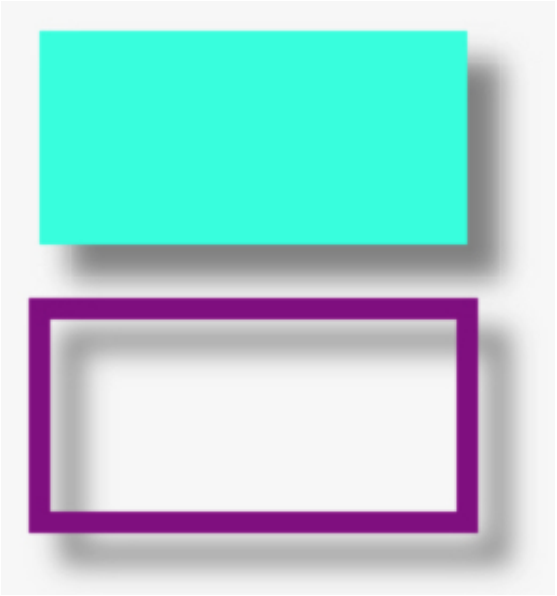
CONTEXT PROPERTIES TO DRAW WITH SHADOWS

There are 4 properties of the canvas context that are useful for indicating that we want to draw shapes with shadows:

1. `shadowColor`: color to use for shadows,
2. `shadowBlur`: blur level for shadows,
3. `shadowOffsetX`: horizontal distance of the shadow from the shape,
4. `shadowOffsetY`: vertical distance of the shadow from the shape

EXAMPLE 1: SIMPLE

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



HTML source code:

```
<!DOCTYPE html>
<html>
<body onload = init();>
    <canvas id="myCanvas" width="400"height =800>
        Your browser does not support the canvas tag.
</canvas>
</body>
</html>
```

JavaScript source code:

```
var canvas, ctx;
function init() {
    canvas =document.getElementById('myCanvas');
    ctx = canvas.getContext('2d');
    // call to a function that will set the 4 context
properties for shadows
    setShadow();
    // all drawings that will occur will cast shadows
    // first green filled rectangle
    ctx.fillStyle = "#22FFDD";
```

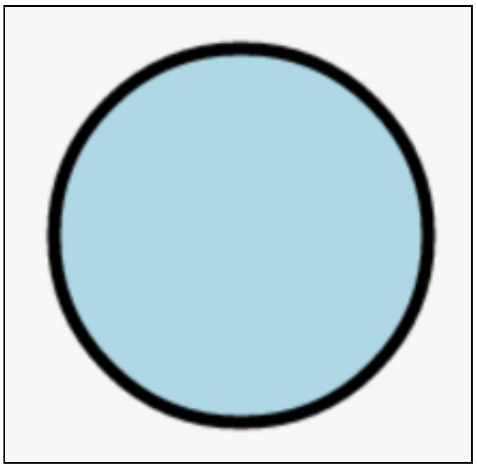
```
        ctx.fillRect(20, 20, 200, 100);
        // second stroked rectangle
        ctx.strokeStyle = "purple";
        ctx.lineWidth=10;
        ctx.strokeRect(20, 150, 200, 100);
    }

21. // We define the 4 properties in a dedicated function, for
    clarity
22. function setShadow() {
        ctx.shadowColor = "Grey";     // color
        ctx.shadowBlur = 20;          // blur level
        ctx.shadowOffsetX = 15;       // horizontal offset
        ctx.shadowOffsetY = 15;       // vertical offset
    }
```

- *Lines 21-27*: we set the 4 properties that define shadows in a dedicated function, for better clarity.
- Line 8: we called this function once before drawing the rectangles.
- *Lines 11-18*: we draw a filled and a stroked rectangle. Both rectangles cast shadows.

EXAMPLE 2: UNWANTED SHADOWS!

Let's take a previous example: the one that draws a filled circle with an outline: http://jsbin.com/gazuba/2/edit



Now, let's add a shadow to it, online example: http://jsbin.com/gokemu/1/edit

Here is an extract from the code:

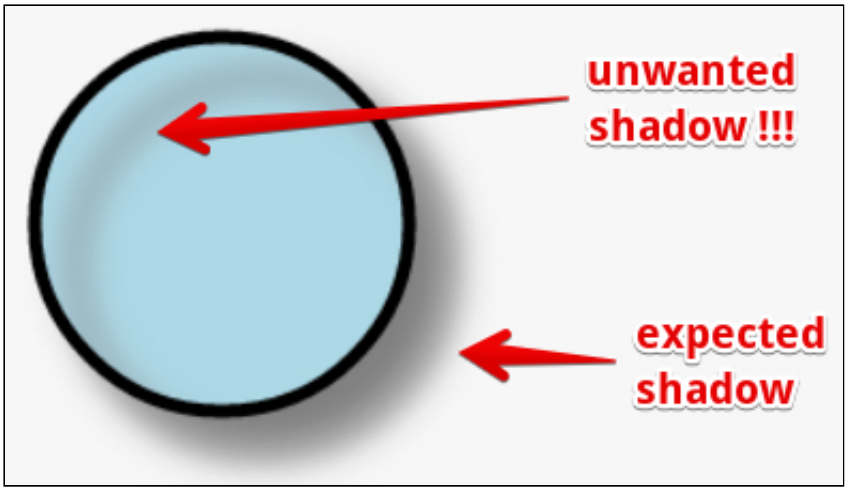
```
...
ctx.beginPath();
// Add to the path a full circle (from 0 to 2PI)
ctx.arc(centerX, centerY, radius, 0,2*Math.PI, false);
// With path drawing you can change the context
// properties until a call to stroke() or fill() is performed
ctx.fillStyle = "lightBlue";

// add shadows before drawing the filled circle
addShadows();
// Draws the filled circle in light blue
ctx.fill();
// Prepare for the outline
ctx.lineWidth = 5;
ctx.strokeStyle = "black";

// draws the path AGAIN (the circle), this
// time in wireframe
ctx.stroke();
// Notice we only once called context.arc()! And drew it
twice
// with different styles
...

function addShadows() {
    ctx.shadowColor = "Grey"; // color
    ctx.shadowBlur = 20;      // blur level
    ctx.shadowOffsetX = 15;   // horizontal offset
    ctx.shadowOffsetY = 15;   // vertical offset
 }
```

And here is the result:



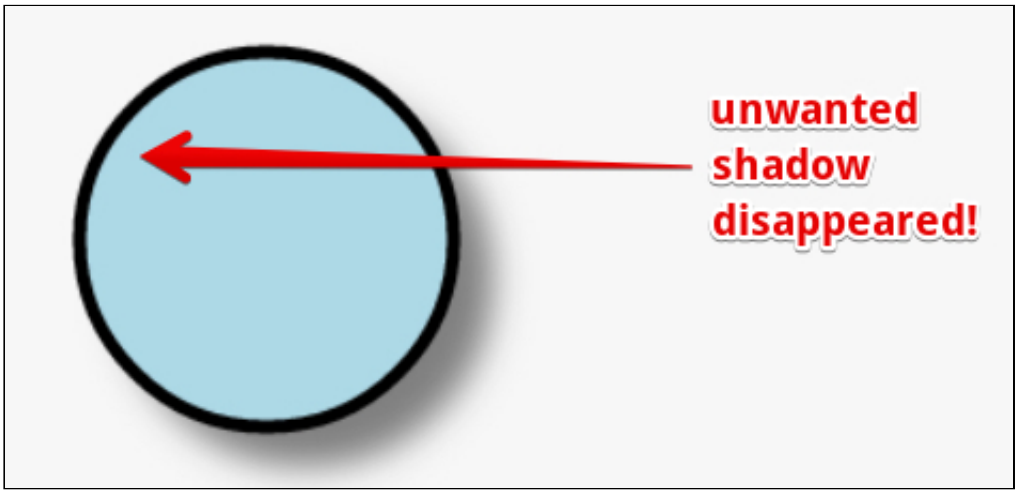
Ah, indeed, the call to `ctx.fill()` casts a shadow, but the call to `ctx.stroke()`, that paints the whole path again, casts a shadow too, and this time the outline produces an unwanted shadow... How can we avoid this effect, while using the same technique for drawing the path?

The trick is to save the context before setting the shadow properties, then draw the filled circle, then restore the context (to its previous state: without shadows), then draw the outlined circle by calling `ctx.stroke()`.

Correct version of the code: http://jsbin.com/kedobi/2/edit

```
...
 // save the context before setting shadows and drawing the
filled circle
 ctx.save();
 // With path drawing you can change the context
 // properties until a call to stroke() or fill() is
performed
 ctx.fillStyle = "lightBlue";
 // add shadows before drawing the filled circle
 addShadows();
 // Draws the filled circle in light blue
 ctx.fill();
 // restore the context to its previous saved state
 ctx.restore();
...
```

And here is the final result:



KNOWLEDGE CHECK 3.5.6 (NOT GRADED)

Shadows are set using the `strokeStyle` or `fillStyle` property of the context?

- ○ Yes
- ○ No
