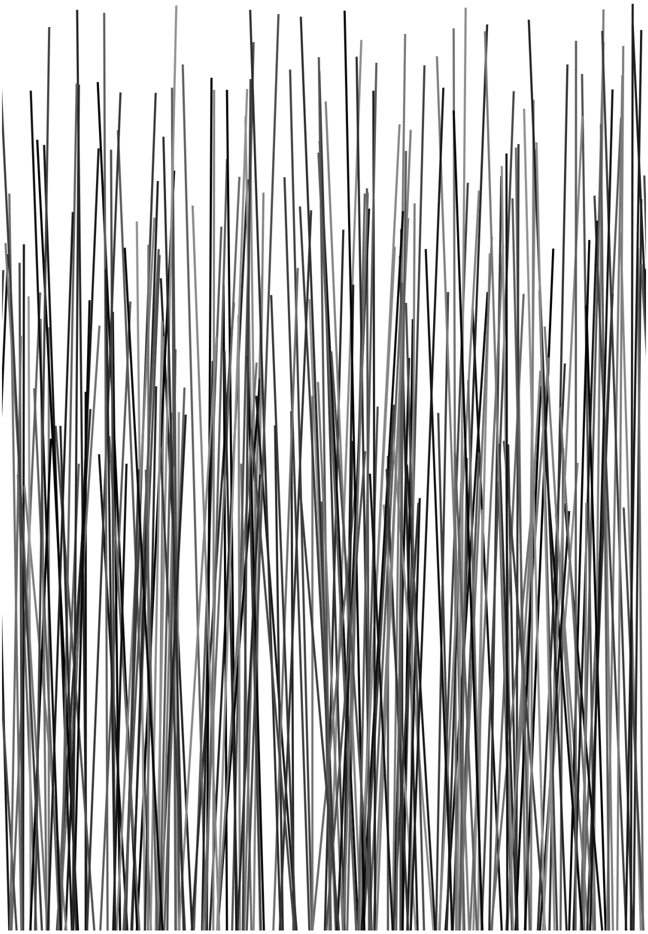
Splash of Code

Blades of Grass

Learn JavaScript by Making Computer Art



Joel Dare

Foreword

I believe the best way to learn programming is by typing someone else's code. You'll have to read it all, in order to type it, and you'll probably make a few mistakes. Then you'll have to learn how to spot and correct errors.

This is a short book. It's short for a few reasons. First, because you'll find that copying someone else's code, although not difficult, takes quite a bit of time. You'll probably have a few challenges the first time, but re-check the code and try again and it will become easier. Second, It's short because I like reading short books and hope you will too. Like you, I'm busy doing things I love. I work full time and have dozens of hobbies that keep me busy. I hope you'll be glad this book gets to the point quickly. Third, this book is created in the spirit of “zines”. Zines are short small-circulation self-published booklets. They’re an art form that I’ve discovered and love.

In each book you'll make an art piece by typing code from these pages into your computer. No previous experience, special tools, or skills are necessary. Along the way you'll start to absorb some computer programming techniques. In the end you'll create an art piece that's fun to display and discuss.

You’ll be working in JavaScript, a language that works in your web browser. If you’d like to go a little deeper, read *How to Code* for a quick introduction to JavaScript programming. It provides a basic understanding of the JavaScript console, data types, variables, math, and functions. You should find the background information there helpful and interesting.

Introduction

Our goal is to create a simple, abstract, image inspired by the plush green lawns of summer. I'll print mine from an old black and white laser printer on standard paper. If you want to print yours, any printer will do.

Remember that you don't need to understand everything you're entering. Just relax and copy the code.

Getting Started

You'll need a computer with a web browser. Nothing else is required. I've created a basic HTML file for you on *CodePen*, a tool that lets you write code online. Open the URL below to get started.

```
https://codepen.io/codazoda/pen/aPVGGx
```

This book includes both a *Complete Code* listing and a *Code Walk Through*. The walk through explains each of the small sections in more detail. You can start at either section. If you want to get right to work, start with the *Complete Code*. If you'd rather read a little more about it, start at the *Code Walk Through* and read it all the way through. Once you're ready to start, turn to the *Complete Code* section and retype the code into CodePen.

The project consists of two files. An HTML file that you won't need to edit and a JavaScript file. You'll retype the code into that JavaScript file.

As you work on this project you'll probably run into typos that create errors. If the image doesn't draw at all, or in a way that you expect, double check that you typed everything correctly and then look at the output in the developer console to see if there are any errors. Because each browser is different, you'll want to search the internet for instructions on how to open the developer console in your particular web browser.

Once you've created an image that you're happy with you can save it by right clicking on the image and selecting *Save Image As*.

Complete Code

```
// Setup some params
var width = 4 * 300;
var height = 6 * 300;

// Grab the canvas element from the page
var canvas = document.getElementById("myCanvas");

// Set the canvas width and height
canvas.width = width;
canvas.height = height;

// Grab the 2D context of the canvas to draw on
var ctx = canvas.getContext("2d");

// Loop a bunch of times
var blades = 250;
var i = 0;
for(i=0; i<=blades; i++) {
      // Pick a random starting location
      let startX = rand(-50, width+49);
      let startY = rand(height*.10, height*.50);
      // Pick a color
      let gray = rand(0, 150);
      // Draw the blades
      grassBlade(startX, startY, gray);
}

// Draw a blade of grass
function grassBlade(x, y, g) {
      let offset = rand(0, 100);
      let direction = rand(0, 1);
      let grays = g + ", " + g + ", " + g;
      let rgb = "rgb(" + grays + ")";
      ctx.beginPath();
      ctx.strokeStyle = rgb;
      ctx.lineWidth = 5;
      ctx.moveTo(x, y);
      if (direction === 0) {
      ctx.lineTo(x-offset, height);
      } else {
      ctx.lineTo(x+offset, height);
      }
      ctx.stroke();
      ctx.closePath();
}

// Pick a random number
function rand(min, max) {
      max = max + 1;
      let rand = Math.random();
      return Math.floor((rand * max) + min);
}
```

Code Walk Through

In this section we'll discuss how the code works but we'll jump around a bit. Read this section all the way through and use the *Complete Code* listing as your reference when typing the code.

The first thing we do is set up some basic page parameters. We want the finished image to be 4 inches wide and we'll multiply that by 300. That will give us 300 pixels, or dots, per inch on the printed page. We'll also set the height to six inches and multiply that by the same 300.

```
// Setup some params
var width = 4 * 300;
var height = 6 * 300;
```

Next we grab the canvas element from the html page and assign it to a variable called *canvas*.

```
// Grab the canvas element from the page
var canvas = document.getElementById("myCanvas");
```

Now that we have our canvas element we need to set it's width and height to those variables we created initially.

```
// Set the canvas width and height
canvas.width = width;
canvas.height = height;
```

In order to draw on that canvas we need to grab it's context and we'll set that in a variable called *ctx*.

```
// Grab the 2D context of the canvas to draw on
var ctx = canvas.getContext("2d");
```

We're going to need to generate a bunch of random numbers. The way you pick random numbers in JavaScript is a bit lengthy so we're going to write a short *rand()* function to make our code a bit more concise.

```
// Pick a random number
function rand(min, max) {
    max = max + 1;
    let rand = Math.random();
    return Math.floor((rand * max) + min);
}
```

I typically organize my functions at the bottom of my JavaScript code. As such, you'll want to move your cursor back up above the *rand()* function you just wrote and continue from there.

Now we're almost ready to draw something.

We're going to draw a bunch of lines that represent blades of grass. To do that, we'll create a loop and generate 250 blades.

```
// Loop a bunch of times
var blades = 250;
var i = 0;
for(i=0; i<=blades; i++) {
    // Looped code goes here
}
```

Each pass through this loop we're going to pick a random starting point and then draw a line from that point to the bottom of the canvas.

Pixel coordinates are specified as *x* and *y*. The *x axis* is the horizontal position on the page starting at zero and the *y axis* is the vertical position on the page also starting at zero. That means that *0,0* is the pixel at the top-left corner of the page.

We're going to start our line, for each blade of grass, at any position on the *x axis* up to 50 pixels off the canvas. Because the left side is zero, we'll allow it to start anywhere from -50 to the full width of the page plus 49. In other words, our random number will be between -50 and the canvas width + 49.

For the *y axis* we'll want each blade to start in the top half of the page but not in the top ten percent of the page. So, we'll pick a position on the *y axis* anywhere from ten percent down the page to 50 percent down the page.

```
    // Pick a random starting location
    let startX = rand(-50, width+49);
    let startY = rand(height*.10, height*.50);
```

We're also going to draw each blade in a slightly different shade of gray. As a result we need to pick a random gray amount between 0 and 150.

```
    // Pick a color
    let gray = rand(0, 150);
```

We're going to make a function for drawing each blade of grass. That function will take an *x* and a *y* argument for those starting locations we talked about, as well as a *g* argument for the gray color. We'll put this function just above the *rand()* function we covered earlier.

```
// Draw a blade of grass
function grassBlade(x, y, g) {
    // Function code goes here
}
```

For each blade we're going to select a random offset. This offset will be used for how far the blade leans. We'll also select a direction value for the direction it will lean.

```
    let offset = rand(0, 100);
    let direction = rand(0, 1);
```

Next we'll set up an RGB string to set the color for the grass blade. Because it's gray, we'll repeat the value three times and concatenate that into a string.

```
let grays = g + ", " + g + ", " + g;
let rgb = "rgb(" + grays + ")";
```

Now we'll start the blade. We'll call the *beginPath()* method, set the stroke style to the rgb value we created above, and set the line width to 5 pixels. Finally we'll move the cursor to the *x, y* starting position for this blade.

```
ctx.beginPath();
ctx.strokeStyle = rgb;
ctx.lineWidth = 5;
ctx.moveTo(x, y);
```

If the direction of the offset is 0, we'll have the blade lean to the right. In that case, we'll subtract the random offset we picked earlier and draw a line to that offset and then to the very bottom, the height, of the canvas. Otherwise, we'll have the blade lean to the left, so we'll add the random offset to x and draw that line to the bottom of the canvas.

```
if (direction === 0) {
    ctx.lineTo(x-offset, height);
} else {
    ctx.lineTo(x+offset, height);
}
```

Finally, we'll make the stroke that we've defined and close the path.

```
ctx.stroke();
ctx.closePath();
```

Now that we've got a function for drawing a blade of grass we'll move back into our loop and draw each blade of grass.

```
    // Draw the blades
    grassBlade(startX, startY, gray);
```

That’s it. Now we should be able to refresh the page and see how it looks. Because we’re using random positions the image will look different each time it’s drawn. When you’re happy with the result you can save it by right clicking on the image and selecting *Save Image As*.

More Reading

Splash of Code is a complete series of short zines. You’ll find additional issues on our website at the address below.

```
https://splashofcode.com
```