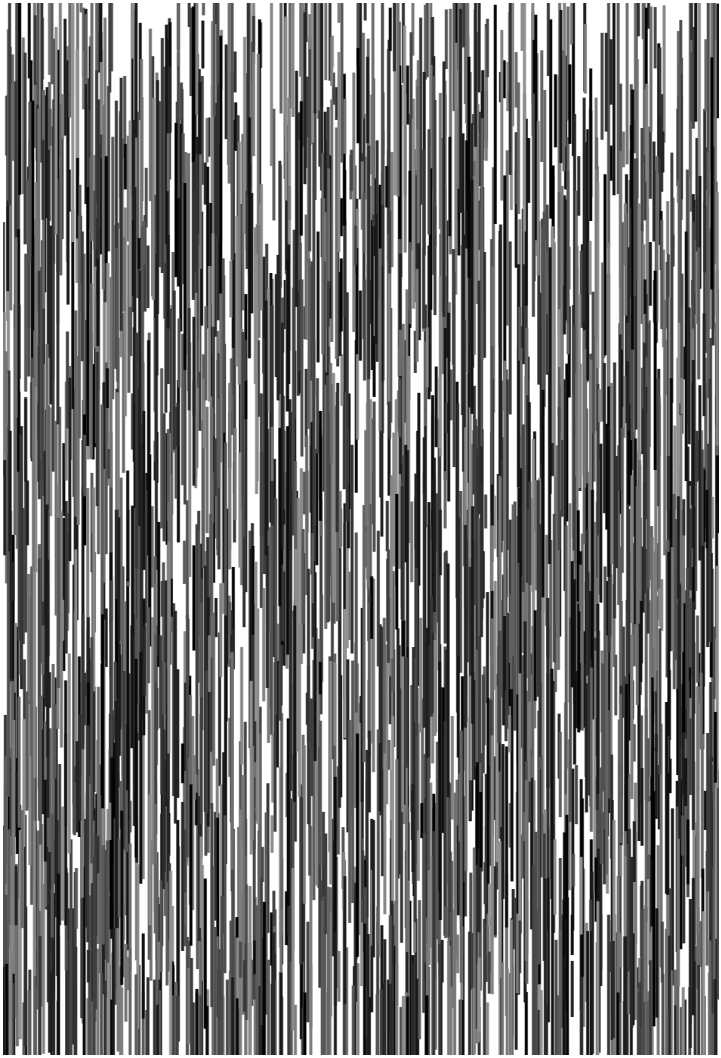
Splash of Code

Falling Rain

Learn JavaScript by Making Computer Art



Joel Dare

Forward

Years ago, I created a software program called ButtonWiz. I gave it away in magazines and on the Internet. When you started the program it would ask you to pay a small fee if you liked and wanted to continue using it. In its heyday it was downloaded nearly a thousand times a month and about 5% of those people paid for it. It was exciting and it was a fun little side project for me.

As ButtonWiz started to pick up steam my employer asked me to focus on his small company. I could see potential in the company and thought I might be a big part of it someday. I stopped making ButtonWiz and focused on my work.

A handful of years later I left that job for another.

Although it was a small success, I've missed the community that formed around ButtonWiz, the feedback I received and the joy it brought people.

As you learn to code, I hope you create amazing things that bring you as much joy as my code brings me.

Introduction

Splash of Code is a series of short guides that walk you through the steps for creating artwork using computer programming. We'll be working in JavaScript, a language that works in your web browser with no additional tools.

A very basic understanding of JavaScript is helpful but not completely necessary. Read my guide, *How to Code,* for a quick introduction to JavaScript. That will help you with a basic understanding of the JavaScript console, data types, variables, math, and functions. You may find the background information there helpful.

Our goal is to create a simple image inspired by the heavy rain of a summer night. 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.

I've provided the complete source code for this project so you can retype it from the code listing here. By doing so you should end up with a finished piece of art that's unique and that you can print, frame, and display. You'll also start to learn some

JavaScript programming. You don’t need to understand everything right away. Just relax and copy the code. You’ll be creating great artwork and you should start to learn a few things as you go along.

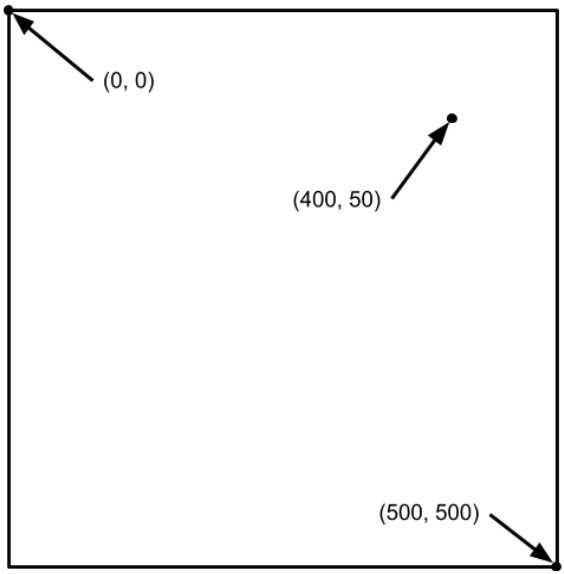
Learning to code requires a lot of patience and repetition. You won't understand everything by creating this one project. In fact, your first few projects may be more frustration than enlightenment. Keep trying. The learning comes over time.

If you dedicate yourself to creating a dozen projects, you'll start to understand the key concepts. The more you search, study, and evaluate, the more you'll learn. Before you know it you'll find yourself considering improvements and alternatives for the code you're reading and writing.

The Coordinate System

When we draw on a computer we typically use a two dimensional coordinate system. The screen acts as a plane and we specify points on the plane using a set of numerical coordinates. Imagine laying a ruler across the screen. The position side to side is called the *x axis*. Now imagine turning the ruler so it measures up and down. This is called the *y axis*.

Each point is specified by a pair of numbers *(x, y)*. By default, the system starts at the top-left corner of the drawing. That point is specified as *(0, 0)*. The point *(400, 50)* indicates *400* pixels from the left and *50* pixels from the top.



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

Start at the *Code Walk Through* and read it all the way through. There I'll explain each of the small sections in some detail. Don't worry if you don't understand most of it. Understanding it comes with repetition. Once you've read it over turn to the *Complete Code* section and re-type the code into CodePen. You can also go back to the *Code Walk Through* for a little overview of anything you want more information about. Then search the internet for more detailed explanations.

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 in this guide 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, or doesn't draw in the way you expect, double check that you typed everything correctly and look at the output in the developer console to see if there are any errors. Search the internet for instructions on how to open the developer console in your particular web browser.

Once you've created an image, you can save it by right clicking and selecting *Save Image As*.

Code Walk Through

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'll grab the canvas element from the html page. We use this to refer to an object on the page where our image will be drawn. A lot of our other commands will draw on this canvas element.

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

By default the canvas is set to a width of 300 and a height of 150. We want to adjust that so we set the width and height to the values we calculated above.

```
myCanvas.width = width;
myCanvas.height = height;
```

The canvas element has a "context", which we'll use later for drawing on the canvas. Here, we assign that context to a variable called *ctx*.

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

When rain drops through the air it creates a downward streak. We want to loop a bunch of times drawing these streaks of rain. Each time through the loop we'll draw a short vertical line. We'll pick a random location, length, and grayscale color for each line and then use those parameters to draw that line.

```
// Loop a bunch of times
for(i=0;i<=2500;i++) {
      // Pick a random location
      let startX = rand(1, width-1);
      let startY = rand(-75, height-1);
      let length = rand(50,400);
      // Pick a color
      let gray = rand(0, 150);
      // Draw the streak of rain
      rainDrop(startX, startY, length, gray);
}
```

Now we need to write a couple of functions. A function is kind of a subprogram that can be run over and over again by other code. Usually, we write these functions just before we need them. I like to organize my functions at the end of the program. That's just a convention I prefer to follow. Other programmers might put them at the top or

sometimes even in the middle. Since you'll be typing this code in order, you'll end up writing them last.

In the previous section we used a function called *rainDrop()* to draw each streak of rain. That's a custom function that draws the actual line on the canvas context we talked about earlier. We need to write that function and we'll do that here.

```
// Draw a short vertical line and drip it down
function rainDrop(x, y, l, g) {
      var rgb = "rgb(" + g + "," + g + "," + g + ")"
      ctx.beginPath();
      ctx.strokeStyle = rgb;
      ctx.lineWidth = 5;
      ctx.moveTo(x, y);
      ctx.lineTo(x, y + l+1);
      ctx.stroke();
      ctx.closePath();
}
```

Finally, we'll write the *rand()* function that we've used throughout. JavaScript has a *Math.floor()* method that we can use to generate a random number, but it's a very long bit of code that I find just a little confusing. We'll encapsulate it in a function so the code is a little easier to read and write.

```
function rand(min, max) {
      return Math.floor((Math.random() * max) + min);
}
```

That’s it.

Once it's written, 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*..

Complete Code

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

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

myCanvas.width = width;
myCanvas.height = height;

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

//ctx.strokeStyle = "#FFFFFF";

// Loop a bunch of times
for(i=0;i<=2500;i++) {
	// Pick a random location
	let startX = rand(1, width-1);
	let startY = rand(-75, height-1);
	let length = rand(50,400);
	// Pick a color
	let gray = rand(0, 150);

	setTimeout(
	function () {
		rainDrop(startX, startY, length, gray);
	}, i
	);
}

// Draw a short vertical line and drip it down
function rainDrop(x, y, l, g) {
	var rgb = "rgb(" + g + "," + g + "," + g + ")"
	ctx.beginPath();
	ctx.strokeStyle = rgb;
	ctx.lineWidth = 5;
```

```
        ctx.moveTo(x, y);
        ctx.lineTo(x, y + l+1);
        ctx.stroke();
        ctx.closePath();
}

function rand(min, max) {
        return Math.floor((Math.random() * max) + min);
}
```

In *Falling Rain* you'll start to learn programming by creating a piece of minimalist abstract computer art by retyping code from this zine (pronounced zeen).

/