Creative Code Week 5 – Pixels & Animation 2

Author: Liana Bourdon
Course: COM214 – Creative Code (Fall 2025)

Overview

This project builds on the earlier Scratch assignments and converts them into p5.js, a JavaScript library used for creative coding and visual interaction. The goal for Week 5 was to design a small collection of animated sketches using procedural logic and user interaction while demonstrating the fundamental programming constructs covered in class.

Seven unique sketches were created and integrated into a Rails-based web portfolio. Each sketch highlights at least one of the required programming concepts, and together they satisfy all assignment requirements.

Tools and Frameworks

Ruby on Rails 7.2 – Provides the web framework for hosting and routing the sketches.

p5.js (v1.9.2) – A JavaScript library for graphics, animation, and interactivity.

p5.play (v3.33) – An add-on library for physics and sprite movement, used in Sketch 1.

Bootstrap 5.3.3 – Used for responsive page layout and button styling.

Font Awesome 6.7.2 – Supplies visual icons and consistent typography.

Heroku (Ruby Buildpack) – Handles hosting and deployment of the production application.

Git and GitHub – Used for version control and remote backup.

Sprockets (Rails Asset Pipeline) – Precompiles and manages all JavaScript assets for production.

Site Structure

Each sketch is served as an individual HTML template located in app/views/creative_code/weeks/5/sketchX.html.erb.

The controller (creative_code_controller.rb) dynamically renders each sketch based on the week and sketch number provided in the URL. Routes were added to handle the structure /at101/weeks/:week/sketch/:sketch. This allows a single controller method to load any sketch dynamically without manual routing.

All scripts are stored in app/assets/javascripts/ as sketch1.js through sketch7.js.

Individual Sketches

Sketch 1 – Bouncing Ball

A simple animation demonstrating motion, gravity, and collision detection. The ball bounces continuously within the canvas and reverses direction when it hits the edges. The logic uses velocity variables and conditionals to maintain consistent movement. Implemented with both p5.js and p5.play for smoother physics.

Sketch 2 – Keyboard Walker

A small rectangle moves across the canvas in response to keyboard input. The program listens for arrow or WASD keys and adjusts its position accordingly. Conditionals prevent the object from leaving the canvas area. This sketch demonstrates variables, loops, user input, and basic control logic.

Sketch 3 – Teleporter

A circle moves around the screen and teleports to the opposite edge when it leaves the canvas boundary. It uses position wrapping, keyboard control, and simple conditionals to create smooth looping behavior.

Sketch 4 – Abby (Follower)

A visual object named Abby follows the mouse cursor using easing. Each frame, the code adjusts Abby's position slightly toward the mouse position, creating natural motion. This sketch introduces an additional custom function beyond setup() and draw(), as well as mouse input and continuous updating.

Sketch 5 – Random Walker

A particle moves randomly across the canvas, taking small steps in random directions each frame. When it leaves the screen, it wraps to the opposite side. This sketch demonstrates randomness, conditionals, and variable tracking.

Sketch 6 – Rain of Circles

A group of colorful circles fall from the top of the screen, each resetting to the top when it reaches the bottom. A for loop iterates over all raindrops, updating their positions continuously. Speed and color are randomized to create a natural rainfall effect.

Sketch 7 – Interactive Paintbrush

The mouse acts as a digital paintbrush. As the user moves and clicks, colorful strokes appear that vary in size and color. This sketch includes user input, randomness, and dynamic control of color and stroke weight.

Requirement

The assignment required that all of the following programming concepts appear at least once. Across the seven sketches, every requirement is met.

Conditional statements: Used in all sketches (collision detection, bounds checking).

For or while loop: Implemented in Sketch 6 (looping through raindrops).

Custom variables: Used in every sketch (x, y, vx, vy, speed, color).

Function other than setup() or draw(): Implemented in Sketch 4 (Abby follower logic) and Sketch 7 (brush control).

User input: Appears in Sketches 2, 3, 4, and 7.

Randomness: Appears in Sketches 5, 6, and 7.

Technical Changes and Improvements

1. Dynamic Routing – Added a sketch route in config/routes.rb and a new controller action.
2. Asset Pipeline Integration – Each JavaScript file was added to the manifest for Heroku.
3. Local Libraries – Local copies of p5.min.js and p5play.js were added for offline reliability.
4. Navigation Enhancements – Added previous/next and week navigation buttons.
5. Responsive Layout – Centered the canvas and buttons using Bootstrap.
6. Download Feature – Each sketch includes a download link to its JavaScript file.
7. Heroku Deployment – All assets precompiled and fingerprinted for production.

Testing and Validation

All seven sketches were tested locally and on Heroku. Each rendered correctly with animations, input, and layout. User interactions were tested with both keyboard and mouse, and random elements behaved as expected.