Timna Costa Aversa

Introduction to Programming ll

10 March 2025

Drawing App Case Study Report

List of Modifications and Extensions

All the below tools seamlessly integrate into the toolbox, adhering to the established framework with a `draw()` function, a designated name, and an icon. It becomes operational upon user selection.

Shape Tool

The Shape Tool introduces the capability to add predefined shapes to the canvas. The user experience follows a straightforward flow: select the tool, choose the desired shape (rectangle is the default), and drag the mouse to preview the shape's size and position relative to the initial click point. Releasing the mouse click finalizes the shape on the canvas.

The code structure includes a shape selector, defaulting to the rectangle, and a main `draw()` function that calls the appropriate shape drawing function based on the user's choice. The selected shape is dynamically updated as the mouse is dragged and permanently added to the canvas upon mouse release.

Image Tool

The Image Tool enables users to insert images from their local machines onto the canvas, expanding the application.

To utilize the Image Tool, the user first selects it from the toolbox. This activates tool-specific options at the bottom of the application window, allowing the user to choose an image file from their computer. Once the image is selected and its dimensions are set, the user can click on the canvas to place the image at the specified size. Alternatively, the user can drag the mouse on the canvas to dynamically resize the image while maintaining its original aspect ratio.

Text Tool

The Text Tool empowers users to add text to their drawings. Using the tool, the user selects the tool from the toolbox, and a text input field appears. The user can then type in their desired text and use the mouse to position it on the canvas. The code behind the Text Tool manages the text input field, handles the placement of the text based on mouse coordinates, and provides options for adjusting the text size and font.

Line Size and Transparency

This feature provides dynamic control over the line size and transparency for all drawing tools. It is integrated into the color palette section of the application, offering sliders for intuitive adjustment of these parameters. Users can manipulate the sliders to set their preferred line size

and transparency level, which are then applied to subsequent drawing actions. The code retrieves the values from the sliders and updates the drawing parameters accordingly.

Plan Effectiveness

My initial project plan was great, but I encountered some unforeseen difficulties and time management issues. Implementing the Line Size and Transparency feature was complex, as integrating it across all tools proved challenging, and isolating the line size setting from other project settings required cleanup to avoid unintended interactions. The Image tool also presented a hurdle, as dragging initially produced two images: the original size (expected behavior when clicking) and the resized version. Troubleshooting was required to ensure only the final, resized image remained.

Despite these challenges, I successfully implemented all planned tools and extensions, demonstrating flexibility and problem-solving skills. The project's success is evident in its smooth user experience, facilitated by the framework I utilized. Users can easily access all tools, with colors and line controls consistently positioned at the bottom of the page, ensuring a consistent and intuitive interface.

However, there's always room for improvement. Given more time, I would refine the Text tool to allow for dynamic resizing with the mouse. Additionally, I would have liked to add a scissors tool for enhanced image editing capabilities.

Overall, the project was a valuable learning experience. It reinforced the importance of careful time management and highlighted the challenges of implementing complex features. I'm proud of the final product and the skills I developed. One key takeaway is recognizing that not

all tools require the same effort, and time allocation should reflect the complexity of each component. This learning will be crucial for future projects and ensuring efficient time management.

Project Evaluation

This project aimed to enhance the existing drawing application by implementing new tools and improving the overall user experience. The evaluation focused on code modularization, readability, and the usability of the application.

Several new tools were successfully integrated into the application, including shape and image tools, along with controls for line size and transparency. These tools expanded the creative possibilities for users while maintaining a simple and intuitive interface. The application's user-friendliness was a key strength, making it accessible to a wide range of users.

However, due to time constraints, the project has some limitations. The range of brushes and shapes is currently restricted, and advanced features like a scissors tool and layering functionality are not yet available. Additionally, the text tool's user experience could be further refined to provide a smoother and more intuitive text input and positioning process.

Despite these limitations, the project successfully delivered on its core objective of enhancing the drawing application with new tools and an improved user experience. Future development could focus on expanding the toolset, incorporating advanced features, and further refining the usability of existing tools like the text tool.

External Resources

References

1. p5.js Examples: Form - Star. (n.d.). Retrieved from https://archive.p5js.org/examples/form-star.html
2. Yuming, Y. (n.d.). Draw a five-pointed star with p5.js. Retrieved from https://editor.p5js.org/yonghuming/sketches/H1UeBFeF7
3. Coding Train. (2018, May 29). P5.js Tutorial for Beginners - 4: Adding Images [Video]. YouTube. https://www.youtube.com/watch?v=NcCEzzd9BGE&list=PLRqwX-V7Uu6bI1SlcCRfLH79HZrFAtBvX&index=4
4. Coding Train. (2018, May 31). P5.js Tutorial for Beginners - 5: Adding DOM Elements [Video]. YouTube. https://www.youtube.com/watch?v=587qclhguQg&list=PLRqwX-V7Uu6bI1SlcCRfLH79HZrFAtBvX&index=5

AI Assistance: AI was instrumental in the support of providing documentation.

External Libraries: The project relies on the p5.js library for core drawing functionality.