Rachel Simms
Chad Fields
Evan Pietraniec
Bradley Houghton
Aaron Romain

Final Project

12/18/2013

DEARBORN
COLLEGE OF ENGINEERING
& COMPUTER SCIENCE

1. Statement of the problem

The goal is to figure out how to use Github so that an entire class can collaborate on one project. One common library of SVG functions must be used to ensure all students' code is compatible with the others. Each student will be able to add their own graphic function to the library and access the code to create a poster consisting of all student's graphics. The poster will consist of images of all sizes, some bigger than others depending on users preferences.

2. Description of Solution

Our first objective was to learn to use the file sharing client, Github. Github is incredibly popular in the programming community, and has become more of a social network than a file sharing client. Users can help others develop programs, and even seek help for their own. The main benefit of using Github is that nothing is lost when updates are made. This ensures that if an error is made, the file can be rolled back to a previous version from before the error occurred. Projects are stored in what is called a repository, and these repositories can be set to public or private, along with many other customizable features.

First, our program consists of the basic structures that hold data for SVG elements including fill, stroke, line, rectangle, and circle. Each student is able to access these structures and utilize them in the basic SVG drawing functions that are also included in the repository. This allows students to create their own more complex drawing functions, while maintaining compatibility with other students and the main program to compile them all into a poster.

We began by writing a function to create an n x m matrix, and another to set every value of that matrix to zero. This is the framework of our poster. Each space of the matrix accounts for one cell in the poster. In order to make sure images do not overlap and are drawn in the correct space, each time a drawing is about to be made another function checks to be sure its corresponding element in the position matrix is zero. If it is, the image is drawn and the zero in that position in the matrix is changed to a one. The checking, drawing, and position matrix updating functions are all built to accommodate larger images in case an image is going to be larger than the others.

Next, each student will be assigned a number between zero and the class size. Each student will then add their getData, drawImage, and main functions. Students' main functions are already named so that they can be called by our main drawImage function. In this function, each student has a case which calls their graphic information.

Our main contains nested "for" loops to cycle through all the positions in the matrix. This ensures every position is checked so that a drawing may be created there. Each position calls the position checking function to ensure something else, like a larger image in a previous row, has already been drawn there. If not, the next student's image is drawn.

The uniqueness of our code is that it does not actually draw anything. It simply allocates space in which users can place their drawing information. It can accommodate up to 100 students' drawing functions the way it is currently set up. If any more need to be added, the programs we used to write the repeating code in the switch cases and the student code areas are in the repository, and can be used for any positive integer value.

3. Testing and Results

We used our code from earlier quizzes to make sure our program works. As it stands, it accomplishes our original goal. Everything we made is accessible on Github, and any added user can add their own code to or modify the project. It also allows a user to decide how large each image will be drawn, and ensures that nothing will overlap. A sample of the produced image can be seen on the next page.