Ivan Faucett and Saad Chaudry

CIS 1968 Final Project Report

April 23, 2024

For this project, we wanted to develop an instrument in the Java programming language. Our initial idea was to create a virtual instrument, but we then thought about other music-related programming projects after discovering the javax.sound.midi package. Some ideas included building a MIDI sequencer to control an external synthesizer and creating a rudimentary synthesizer. However, after some research, we came across a video of someone who created a simple piano in Java. This peaked our interest because it is very similar to our initial idea of creating a virtual instrument. The source code for the piano was included in the video's description, and we used this as a framework to develop our own instrument around. We wanted to heavily expand the piano program so that it was both unique and more powerful. Our goals for the project were to create a virtual piano instrument that expanded the keyboard beyond a single octave and allows the user to switch between different sounds.

Our program heavily utilizes the java.awt, java.swing, and javax.sound.midi packages. The AWT and Swing packages allowed us to build an intuitive and aesthetically pleasing graphical user interface to control the instrument. After launching the program, three distinct groups of buttons are immediately present to the user. In the middle of the panel, 36 buttons are arranged in the shape of a traditional piano keyboard. The keyboard starts at C, and allows the user to play notes up to B two octaves above the first C. When one of the keys is pressed, a sound is played at a pitch corresponding to the key that was triggered. The note sounds for 250 milliseconds then shuts off. This decision was made to give the sounds some sustain, but prevent too long of an overlap. Above the keyboard, 7 square buttons labeled 0 through 6 are present.

These buttons allow the user to transpose the entire keyboard up or down 12 semitones, or one octave. The program launches with the octave set at a value of 3, which makes the far left C have a MIDI value of 36, so it will trigger a C2 note. Each key has a MIDI value of one greater than the key to its left, then that MIDI value is added by a value of 12 multiplied by the octave value, so the key will sound at the correct pitch. The bank of buttons at the far left of the panel give the user the option of triggering different instrument sounds. Essentially, when one of these buttons is pressed (for example the “Grand Piano” button), the MIDI program change value is changed according to the sound’s General MIDI program change (PC) value. “Grand Piano” has a PC value of 1, “Elec. Piano” has a PC value of 5, “Acou. Guitar” has a PC value of 25, “Organ” has a PC value of 19, and “Synthesizer” has a PC value of 63. These are the basic functions of the program; all buttons use Swing’s JButton. The performance of the program is as expected, there is minimal input latency, and every button works as described without any issues.

The GUI uses the Swing package mostly for creating the various components while the AWT package is used for colors and painting the background the pink-orange gradient color. The main class FinalProject extends JLayeredPane. Originally, the class extended JPanel as the keyboard was drawn through the paintComponent() method. The paintComponent() method was changed to paint the window's background. The keyboard was eventually changed to having buttons which led to the change from JPanel to JLayeredPane as the buttons needed to overlap. All other components were added straight to the main class’s window. A main method was used to create a JFrame. The main class created an instance of itself to inhabit the JFrame.The main method also uses the SwingUtilities.invokeLater() method so the program runs on a new thread as Swing is not thread-safe by itself. Three sections of buttons were created for this program. The buttons at the top let the user choose the keyboard’s octave. The keyboard is made of buttons as

well. The buttons on the left side allow the user to choose between several different instrument options. Each component in the window was placed using x and y coordinates. The components were also put in their separate layers when necessary.

In conclusion, we are very pleased with how the program turned out. We feel that the GUI is simple, intuitive, but most importantly, it is effective. Also, we were able to fulfill all of our initial goals for this project. The program allows the user to play a virtual piano instrument, transpose the notes up or down an octave, and lets the user select between multiple different sounds. The topic of music software and technology is extremely interesting and intriguing to both of us, and we are glad to have designed a program in this subject. If we wanted to develop this program further, we could implement a MIDI file loading feature, expand the instruments available to the user, and allow for external MIDI control of the instrument.