Carolyn Lober

The Clavier1

May 4, 2015

## **Functional Overview**

My project will be an interactive piano using the Melody and related classes from Assignment 12 with the following features:

* Plays notes when the mouse clicks on them
* Registers the keys of the keyboard as notes
* Plays melodies from input files (like Assignment 12) and keys on the piano light up when played
* “Records” and plays back a song played either from mouse clicks or from the keyboard

Possible features, if I have time:

* Store melodies and recordings as MIDI files to allow multiple notes at once

While anyone can use this program, it will probably mostly be me.

## **Design Overview**

I will use a GUI to get user input. Upon starting running the program, a menu screen will show up, asking the user if they want to play the piano (“play”), load/play a song (“load”), record their own song (“record”), or quit the program (“quit”). Any option other than quit will bring up a single octave keyboard:

If the user closes the menu screen, that will also quit the program. Closing any other window will bring the user back to the menu screen.

The user can stop recording or playing by pressing the space bar or closing the piano window.

When a key is pressed, or the computer plays a note in a song, the key will be colored (the shade depends on the note, darker or lighter depending on the octave). While the user is playing, the caps lock key can be used to jump up one octave. NOTE: Only two octaves will be available when the user is playing the piano (middle C octave and the one above) while only three will be available when a melody is playing (one octave below to one octave above middle C). If the user tries to play a melody in the wrong range, the program will notify the user, then go back to the menu screen.

## **Design Details**

I will be using jGRASP as the development environment.

I will use some of the code from the melody assignment, especially the play methods from the Melody and Note classes, and the read method from MelodyMain.

In addition to using Melody and Note objects, I will define a Key object and a Piano object that represents a single octave (what appears on the screen as shown in the diagram above). Each octave will be represented by a different Piano object in the main method.

Class Piano

Instance variables:

List of Key objects—white keys

List of Key objects—black keys

Methods:

Constructor: takes an int parameter that is the octave

Instantiates lists with the keys in the given octave

Default constructor:

Instantiates lists with the keys in the octave with middle C

hitKey: takes int parameters x and y, returns the Key that contains that point

for each black key

if the key has been hit

return the current key

for each white key

if the key has been hit

return the current key

playNote: takes int parameters x and y, plays the note represented by the clicked Key

use hitKey to find the Key that has been hit

play that key

Class Key

Instance variables:

ints rightX, upperY, width, height—the upper left corner of the key and size (for white keys, the area of the key includes the half of the overlapping black key)

a Note object—the note represented by the key

Methods:

Constructor: takes int parameters rightX, upperY, width, height, and Note parameter

Instantiates all instance variables to the given values

Default constructor:

Instantiates all instance variables so that the key represents middle C

isHit: takes int parameters x and y, determines whether this key was hit

if x and y are within the bounds of the key

return true

return false

play

play the note (instance variable)

Menu Screen

Show menu screen

If “play”

Show piano

If user clicks

playNote with location of click

color the key while the note is being played

else if “load”

prompt user for name of file

open, read file (read method from MelodyMain) and save as a Melody

show piano

for each note in the Melody

determine the key that each note will use and store them in a list

for each key in the list of keys

play the key and color the note

else if “record”

prompt user for name of song (file to save to)

while file already exists

check if user wants to overwrite file

if no, prompt for new filename

show piano w/ a button “done”

list of notes -> song

if user clicks

while the click is not the space bar

playNote with the location of the click

color the key while the note is being played

store the note of the key in the song

time any rests and save them in the song

print song with proper formatting to file

if file saving is successful and file is readable

tell user the recording was successful

else

print error message

else if “quit”

quit program

## **Testing**

I will test each feature of my program as I add it.

## **Grading Rubric**

Write your own grading rubric (out of 40 points) that takes into account whether

a) you matched your functional specification (does your program works as planned)

b) is your design a good one? Is it efficient? What does efficient mean in this language or program?

c) is your code well documented and readable?

d) how well was your code tested?

e) what else could you be graded on that isn’t listed here? Dig deep, be creative.

## **Proposed Implementation Schedule**

Pre-coding:

1. Design document will be done by Friday **5/8**
2. Learning about GUI’s (reading BJP chapter 14, writing some simple test programs with what I learn, and more if necessary) will be done by Friday **5/15**

Implementation:

1. I will have an interactive piano that plays each pressed note (both from mouse clicks and the keyboard) and colors the key upon playing it for half a second (I imagine this is easier than playing for however long the click is, but if not, it’s not necessary) by Monday **5/18**
2. I will add a functional menu screen and the feature of playing songs from text files by Wednesday **5/20**
3. I will add the feature of “recording” music by Friday **5/22** or at least by Monday **5/24**
4. Any additional time will be spent changing storage of music as MIDI files and other general improvements

## **Potential Showstoppers**

* List any problems you think could come up in your project.

## **Open Questions**

* Any other questions or concerns?

## **Resources**

List and resources you are currently using or expect to use during the course of this project.