CS361 Project 3
Authors: Phoebe Hughes, Erin Lavoie, Jenny Lin, Joseph Malionek
Date: October 3rd, 2016

In this project, we created a GUI in which the user can click different bars to play notes. The location of the click dictates the pitch and time the note is played. A lower click indicates a deeper pitch, and a click farther to the right plays the note later in the composition. Each note is 100 ticks and represented by a 10 pixel high x 100 pixel wide box. By creating multiple notes, the user can then create a composition and play it using the menu option “Play” or ctrl/cmd + p. When the composition is playing, a red bar appears and moves across the screen. The location of the red bar indicates which notes are currently playing. When the composition is finished the red line will disappear. The composition can also be stopped by using the “Stop” menu option or ctrl/cmd + s. There is also the “Exit” menu option and ctrl/cmd + q shortcut, which allow the user to stop the music and quit the program. We decided to use “Shortcut + <key>” so that on Linux and Windows machines, the shortcuts would be “ctrl + <key>” while on Macs, the shortcuts would be “command + <key>”.

Main
+ start(primaryStage: Stage): void + guiLineSetup(center: Node): void + main(args: String): void

Composition
- BPM: int - DURATION: int - VOLUME: int - midiPlayer: MidiPlayer - noteList: ArrayList<Note> - compositionBox: Pane - redLine: Line - timeline: Timeline
+ exitProgram(): void + handlePlay(): void + stopComposition(): void + addNoteOnClick(click: MouseEvent): void + moveRedLine(stopPosition: int): void + getCompositionBox(): Pane

Note
- pitch: int - startTick: int

Instead of having everything in Main, we decided that there should be a separate class to handle all of the GUI events, which we called Composition. This allowed us to separate the code that initializes the GUI elements from the code that handles the program functionality.

The Main class sets up the GUI and adds all of the lines. We thought that the lines should be added inside the class, instead of in the FXML file, as it prevents the cluttering of the FXML file by reducing code duplication.

Composition contains all of the handler methods for the GUI elements and the reference to the midi player. This reference is necessary so that the handlers can add notes to the midi player and play it. We needed to keep track of the notes added to the composition, therefore we decided to store the note values separately from the gui boxes. The array list storing the notes also allows the reusability of our code by enabling different input methods and prevents problems that arise when we only store the notes in the midi player. In order to facilitate this, we created an inner class Note, which stored the tone and start position of the note (if need be, this could also be extended to include note duration or other attributes). We had to deliberate on where the red line should be handled since it is both part of the layout and part of the functionality of the program. We decided that the more logical place was in the Composition file, because it can mediate between the midi player and the gui much better than the Main file can.