Joshua Bonn
bonn7152@vandals.uidaho.edu
https://github.com/JoshuaBonn1

Synopsis

MuseScore is a free, open-source scorewriter focused on creating, playing, and printing beautiful music. MuseScore has many features to make editing scores of any size easy and intuitive. The current mode of large score navigation is a tool called Navigator. It displays a small version of the visual score that you drag a small box through to change the current visual focus. It is tough to visualize where you are in a large score with a minimized score. That is where the timeline idea comes in.

The timeline would display an abstract view of the score with some key features of the score such as tempo, key signature, and instruments. The abstract view would show when different instruments were playing. To navigate the score, you would select the position on the abstract score and it would display that location.

Benefits to MuseScore

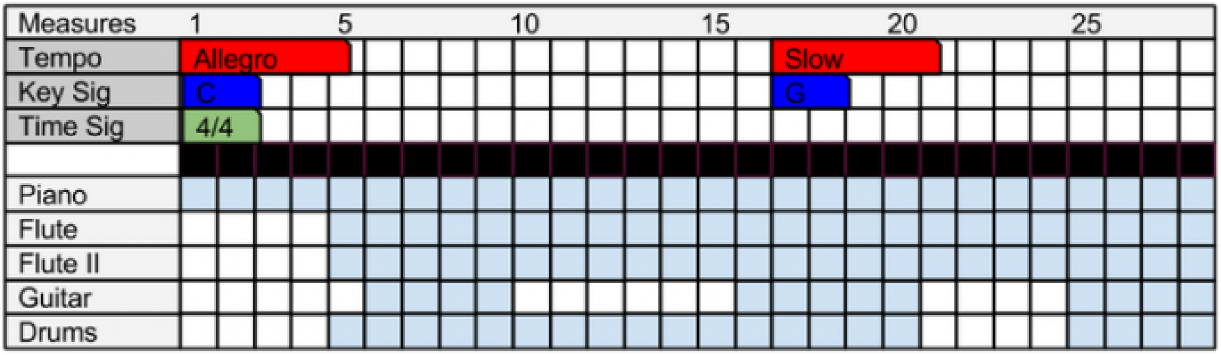
The timeline would benefit the creators on MuseScore to have a quick, comprehensive method to navigate large scores. The current system works well for small pieces, but not as well for large works. It would also aid in the understanding of large works by being a visual assistant of a simple abstraction of the score.

Deliverables

- Integration of the timeline into MuseScore as a tool.
- Full documentation for the above deliverable

Project Details

The timeline would be a grid format as shown below. The grid would display measure numbers, tempo, key signature, time signature, and a list of all the instruments. In the rows for instruments, the boxes will hold a color to indicate that notes exist in the measure. If there are no notes, the square is white. The intensity of the color would increase as the intensity of the measure (measured by notes to number of beats in a measure) increases. If multiple of the same part exist, the user will have the option to collapse the two lines together, such as Flute I and Flute II would become one row. This same feature applies to instruments of the same class, such as woodwinds, brass, strings, etc. The user could add their class of instruments by selecting them and clicking some set of buttons. To navigate using the timeline, the user would click on a square, and the display would shift to the location of the measure. If the square was an instrument, it would display the measure of that instrument. If the square was for tempo, or another main category, it would move to where that change is displayed. For vocal sections, when the mouse hovers over a square, it displays part or all of the lyrics for that section.



Project Schedule

This week-by-week timeline provides a rough guideline of how the project will be done.

12 -- 17 May

Associate myself with the structure of MuseScore, primarily creation and reading of music and creation of tools by using the Developers' Handbook. Practice making small changes to the code and building using Qt Creator. Understand the GitHub workflow used by MuseScore.

17 -- 30 May

Add a new tool for MuseScore named timeline. Using QGridLayout, create a basic timeline, presenting the width as the number of measures and the height as the number of instruments. Implement navigation to the specified measure by clicking the boxes.

31 May -- 6 June

Specifically test the tool with multiple existing pieces on MuseScore and of my own creation. Document my code and plan fixes for any errors in testing that cannot be fixed immediately. If any time is left, begin work on the errors.

7 -- 20 June

Add the time signature, key signature, and tempo to the timeline. Also, add a color change to the boxes where there are notes in the music. Continue careful error checking to avoid large problems.

21 -- 27 June

Test large scores with multiple tempo, key signature, and time signature changes. Handle any graphical issues here. Document all changes.

28 June -- 11 July

Integrate intensity of measures to a darker color for the box. Add the collapsing of multiple similar parts into one. Create simple feature for user specific collapsing. Add tooltips for lyrics in the vocal parts.

12 -- 25 July

Add movement separation. Begin final stretch of testing and documentation.

26 July -- 1 August

Test with complex scores with as many features as possible to stress test the feature.

2 -- 16 August

Further refine tests and documentation for the whole project.

Bio

My name is Joshua Bonn and I am studying at the University of Idaho in computer science. I enjoy making and eating food, coding, and playing Rocket League. I used to also major in Music Theory and play cello. I used MuseScore to rewrite a handwritten concerto that we had in orchestra into a nice typeset edition. I also used the plugins to create an error checker to four part chorales (parallel fifths, tendency tones, etc.). I have written an automatic chorale generator in C using genetic algorithms and other local search methods. I am currently in the process of building an application for my school’s marching band to apply color animation to light up glasses using Qt and C++. Programming and music is a combo I have always loved to mix, from the first personal project I made. My first project was an interval chord, and scale recognition software. I would enjoy working for this project as my first step into open source projects for my personal favorite score writing program.