Heqet Application Report

By Isaac Reilly
advised by Donya Quick
continuation of my senior project for the Yale University major in Computer Science

spring 2016

1 Overview

Last semester, I created the Heqet library to improve music representation and manipulation in Haskell and to export Lilypond code to make pleasing printed scores. My project this semester was to create a graphical application using the Heqet library, with some additional features like multiple selections and a automatically updating dependency graph.

2 Results

The project turned out to be vastly more than I could finish in a semester, but my efforts have laid a solid foundation for future work. The graphical interface, which I built using the `threepenny-gui` library, is almost complete and includes navigation, music rendering, and many palettes of buttons. When it came to rendering the music in real-time, I was hoping to use an speed-optimized form of Lilypond, but this proved impossible. Instead, I created a fast music drawing system which avoided the many aesthetic adjustments of the output which slow down Lilypond, such as varying the horizontal spacing of notes to make both fast and slow sections easy to read.

I adapted the Heqet library for the application and converted it to use the Stack build system for Haskell rather than Cabal. The Heqet library still needs some work, such as implementing dynamics and polishing the time signature handling.

As predicted, the bulk of my work was on the application. I spun off the library for the dependency graph and undo system into the `history-graph` library and built the framework for almost the whole application. I wish to emphasize that although the application is not usable, the code is in a much better state than it may seem. I have solved most of the hard problems of the project, but I ran out of time to implement the large amount of remaining easy features. For example, many of the editing functions corresponding to currently-nonfunctional buttons are trivial, but the interface is laid-out logically and connected to the to the internals. A more complicated example: there is no ability to open or save files, but the file format is straightforward and the necessity of saving files was the cause of an important design point, the function "registry", which provides the ability to encode history actions as text, since functions themselves can't be easily stored in a file. The functional reactive programming UI framework should make it straightforward to add the click-to-select functionality.

3 What I Learned

As in last semester, I've grown in my ability to find satisfaction in a large project that is not complete. I have some time off this summer before starting to work full-time, and I am very excited to continue with Heqet and make it available to users.

My skills in Haskell programming and software architecture have improved, especially when I studied functional reactive programming.

4 Haskell

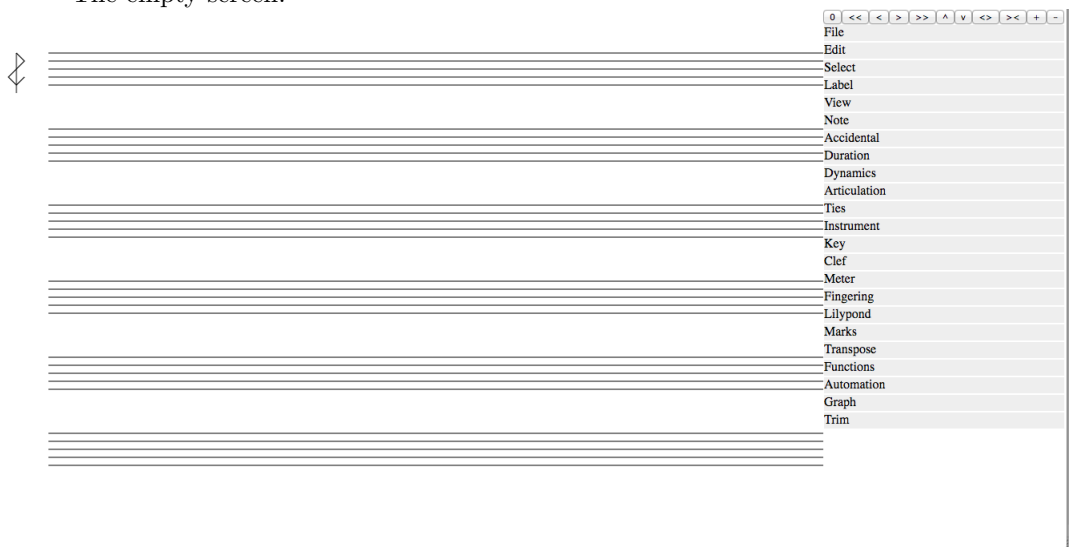
The Haskell language continued to live up to its reputation: I had almost no semantic bugs once clearing the type errors. Refactoring, such as when I added the function registry, went smoothly. My attempts to change the structure of the Heqet library's music representation didn't go well, but that was due to problems with my algorithms which would have necessitated a large refactoring that I decided to forgo in favor of focusing on the application. I didn't know what to expect from functional reactive programming, but it turned out to be logical and concise.

5 Future Goals

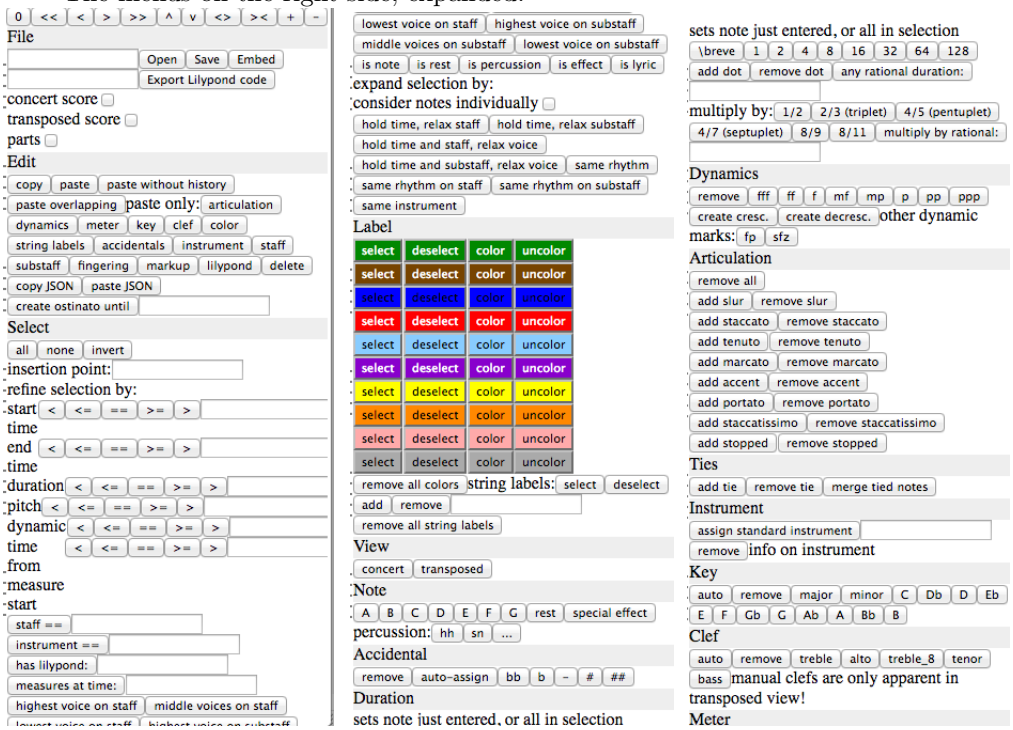
I now have even more open issues on Heqet's GitHub tracker: 149 issues, most of which are unimplemented features. In addition, there are more improvements I would like to make, such as prettier drawn music on the real-time canvas and support for multiple music notation systems, such as a text-based display suitable for screenreaders, braille music in a format for a tactile display, early music notation, and a variety of alternative notation systems including one of my own design. I plan to make complete builds available for a variety of systems. I look forward to continuing with this project!

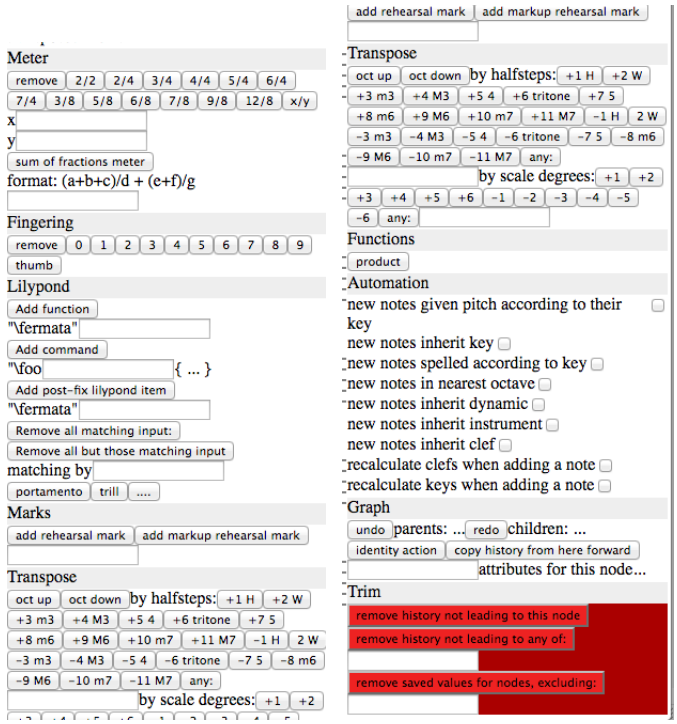
6 Images

The empty screen:



The menus on the right side, expanded:





A demonstration of various glyphs: