question - A questionnaire editor, asker, and parser

Kevin Sun

April 17, 2021

1 Introduction

The application `question` is a piece of software to facilitate constructing and asking questionnaires. It has a TUI for both editing and answering them. This is done by supplying the corresponding arguments when launching the binary. The user can choose to construct a questionnaire using either the TUI or making a .txt file that will parsed by `question`. This file must adhere to the syntax of `question` or the application will not launch. Currently `question` only supports parsing sequential questionnaires, but branching questionnaires can be constructed in the editor TUI and saved to a .txt file. The syntax for branching questionnaires leaves room for future parsing support, but cannot currently be parsed in such a way to reflect the branching paths in the application. The application will simply treat it as a sequential questionnaire, and parse it as such. The same goes for answering questionnaires. The first question in the follow-up questions for some question will always be chosen as the next question to be asked.

2 Features

A question can be open-ended, closed, or closed and mutually exclusive (MU). An open-ended question requires a free form answer, a closed question requires a number of selected options, and a closed MU question requires a single selected option. This is ensured in the answering of a questionnaire in the application.

2.1 Editor

2.1.1 Explorer

This mode enables you to explore the currently loaded questionnaire. Pressing `Up` and `Down` allows the user to scroll between the follow-up questions of the current question. Pressing `Enter` will make the selected question the current question; allowing the user to traverse the questionnaire structure. Pressing `Backspace` will return to the previous question.

Adding an open question `Ctrl+r` allows the user to add an open question. It will open a new window and enables the user to enter a prompt. Pressing `Enter` will stage the currently typed prompt for confirmation and pressing ‘ will confirm it and return the user to the Explorer mode.

Adding a closed question `Ctrl+q` allows the user to add an closed question. It will open a new window and enables the user to enter a prompt. Pressing `Enter` will stage the currently typed prompt and allow the user to enter any number of options to this question. Pressing ‘ will confirm the constructed question and return the user to the Explorer mode.

Adding a closed MU question `Ctrl+w` allows the user to add an closed question. It will open a new window and enables the user to enter a prompt. Pressing `Enter` will stage the currently typed prompt and allow the user to enter any number of options to this question. Pressing ‘ will confirm the constructed question and return the user to the Explorer mode.

Deleting a question `Delete` allows the user to remove the selected question from the follow-up questions.

Saving a questionnaire `Ctrl+s` allows the user to save the current questionnaire to a .txt file.

2.2 Answerer

Answering an open question An open question can be answered by typing in the answer, pressing `Enter` for staging and pressing ‘ for confirmation.

Answering a closed question A closed question can be answered by selecting an option, pressing `Enter` to add it to the answer and pressing ‘ for confirmation. The user can select some other option and also add those to the answer before confirmation.

Answering a closed MU question A closed MU question can be answered by selecting any single option, pressing `Enter` for staging and pressing ‘ for confirmation.

Scrolling The user can scroll in the window using `Ctrl+Up` and `Ctrl+Down` for small scrolling, and `Ctrl+Left` and `Ctrl+Right` for large scrolling.

3 Guide

To compile the app, type `cabal build` in the root project directory. The binary is present in /project/dist/build/project/ and is simply called `project`.

To demo the app, only supply "answer" or "edit". This will load a demo questionnaire. To demo the saving and parsing of questionnaires, make some changes to the demo questionnaire in the Editor and save it. This will spawn a .txt file in the same directory. The name of this file can then be supplied with the desired mode, and edited or answered. Take note that only sequential questionnaires work at the moment, so branching paths constructed in the Editor will not be reflected after supplying it back to the application after saving. You will however see that the grammar for branching paths is already present in the .txt file.

4 Discussion

I feel like I learned a lot from this project. I got a lesson in being too ambitious and underestimating the amount of work that goes into simple features. A big part of making use of libraries is also learning to use these libraries, which also takes time. My coding style evolved significantly, and lenses seem to be pretty interesting. Due to these reasons, a lot of refactoring went on throughout the course of this project. I would like to believe that I made effort to have decent modularity, and I would argue that some of the merits of this project is also present in the organisation of the codebase.