Computer Science 305 - Final Project

Darren Berg, Luke Lupul, Arnaldo Marin

April 11, 2018

Project Overview

Our project was divided into 3 stages, data collection, data display, and data search. For data collection we get our laureate information from the Nobel Prize API, and we use that information to collect images of the laureates from wikipedia. Wikipedia was chosen as the images from the Nobel Prize API were said to be copyright restricted.

Darren Berg

I worked one the database (LaureateDatabase, Prize, Affiliate, ImageHandler, Laureate) and assisted with some of the FXML controllers (tableview, PersonInfoController). Some areas that were difficult for me was in cleaning the JSON (as there are non-people that GSON happily makes) and in finding where I could get images from and collecting them. The names used in the Wikipedia API were very particular and did not line up directly with the laureate names given to us by the Nobel Prize API, I had to iterate different name combinations. Finally trying to handle the load times was a bit of an issues, as we make multiple web requests. The decision for storing the images on the users PC was to help speed up the program (after one initial nasty load time (~3m50sec)) and we were planning on making the program be able to run without an internet connection but didn’t quite finish the code for that feature. A bug in the for images is that the nobel prize image is downloaded preemptively but I couldn’t get it to download using the functions I had. I believe it is due to improper permission on a https request. So if someone deletes that image it won't be making a comeback. My teammates were pleasant to work with, I had a great time working with them. We were able to tackle problems as they came up and meet-up when needed.

Luke Lupul

I worked on the data search part. I designed a class called NarrowSearch which, at construction, built a number of dictionaries (hashMaps) which keyed various categories of data to ArrayLists of Laureates, which were a class the that Darren defined and extracted from the Gson String as taken from the archive. The dictionaries broke up the Laureates by name, and this could be requested from a search bar. It also builds A dictionary where all Laureates are values in a Hashmap whose keys are the decades of the Laureate’s birth or death. I also implemented methods which could reduce such an array to only those alive or only those dead, however these didn’t get implemented at the level of U.I. This is because as is, the amount of information these searches fetch is not overwhelming. If the database were expanded, some auxiliary methods in the NarrowSearch class like this might extend the usefulness of the program if it hypothetically had a life beyond this assignment. Examples of such include search by city. Other Dictionaries key types include the year they won the prize, and the category they won in. Because these dictionaries are built at initialization, that does add a few seconds to the load time, but it improves search time thereafter, so I think it’s justifiable.

Arnoldo Marin

I worked on the GUI prototype, GUI design, and the controllers for the different windows. Darren assisted me in some of the controllers too. I also implemented the CSS files to give visual effects to the buttons. The program is composed of three windows that will appear based on user interaction with the buttons/tableview rows. With the design propose, the user will be able to search by the most important query as well as lookup all the details of every Laureate. The program starts with a splash screen that has an start and an exit button. From there the user can start the program and after a few second, another window will appear where the user can do everything related to the search of the database. The user can search by name in the search bar and he/she also has the ability to search by buttons stored within the drop down menus. If the user double clicks a row with a Laureate in it, a new window will open displaying details of that Laureate. Personally, I found it difficult to set up the different windows based on user actions. I ran into a lot of errors but was able to fix them as I learned more about Java FX. Also, I would constantly run into problems because I would forget to update functions within scene builder but I was able to figure out the problems quickly. We were also experiencing bugs with the searching as if the user would provide bad input, it would crash the program. I was able to fix the bug by adding a try and catch as well as checking for null input and capitalizing the first letter if the user provided an existing name in the database. I had a great experience with my teammates. We were able to meet every time we needed and I had a great time developing this program with them!