Advanced Java Programming

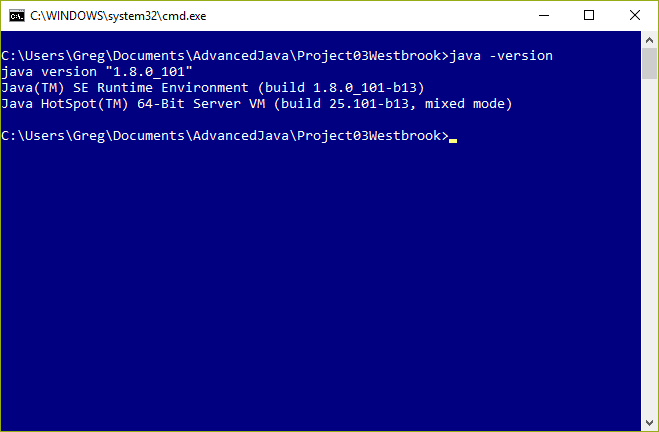
Project07

Due: 2016/11/30

Gregory Westbrook

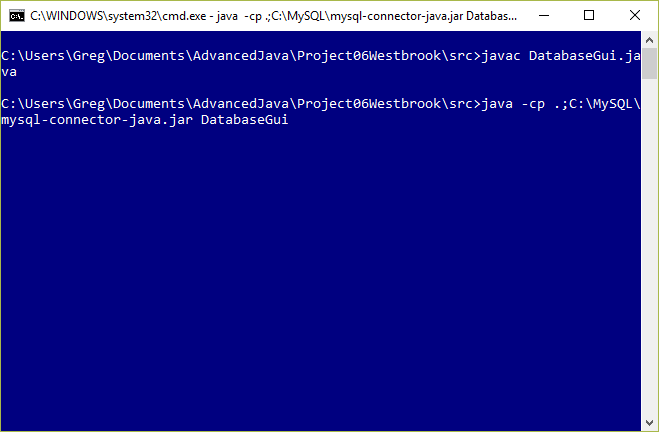
Objective: Create a Java FX GUI program that allows the user to view and edit data from a database using a TableView component.

Here is the version of the compiler I was using:



I was using IntelliJ Community Edition 15.0.3 for the development environment.

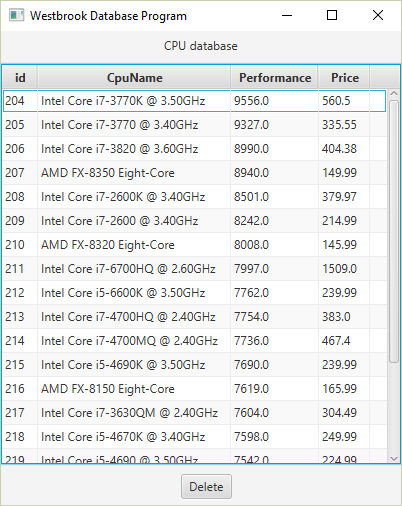
What follows is a screenshot of the program compiling and running.



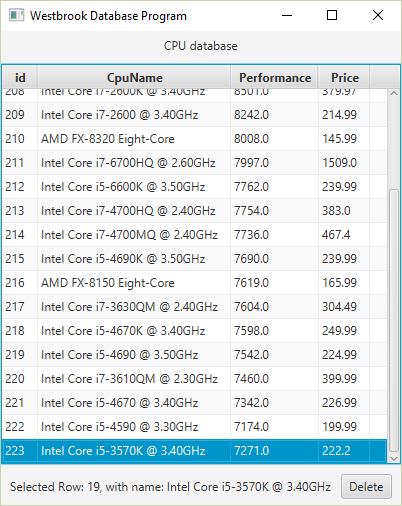
The program uses the Cpu class from Project03 as well as the database left over from that project. The start() method creates an TableView and a label. The action listener on the button is implemented as a lambda and within that, creates the SQL statement that pulls the database contents and displays it in the listView via the getAllCpus() method.

The validation overloads the fromString method of the DoubleStringConverter class and catches the NumberFormatException that is thrown if an invalid number is entered. I stumbled onto this when my application started throwing NumberFormatExceptions and I realized I could just catch those rather than implementing my own regular expression validation which was my original intention.

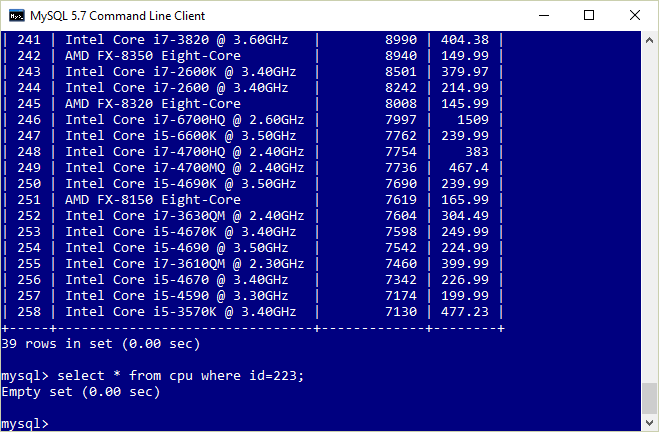
When the JavaFX application starts up, you see the following:



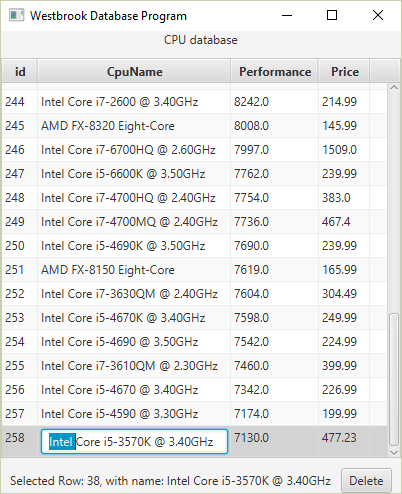
It contains a JavaFX tableView of the contents of the cpu database. This is the same database left over from previous projects and may contain some bogus entries. The ‘delete’ button allows the user to delete those lines one at a time. Simply select the line with the mouse and click ‘delete’ as shown in the following two screenshots.



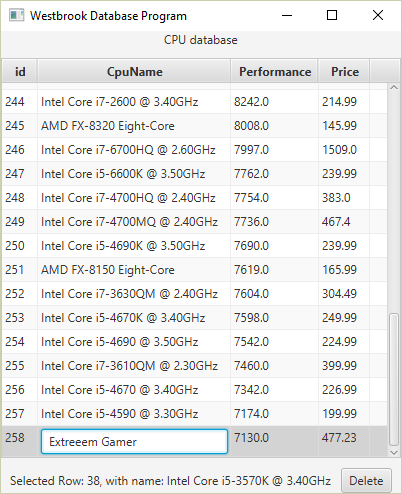
The screenshot below shows a select command from the MySQL command line which demonstrates that the line is gone from the database.



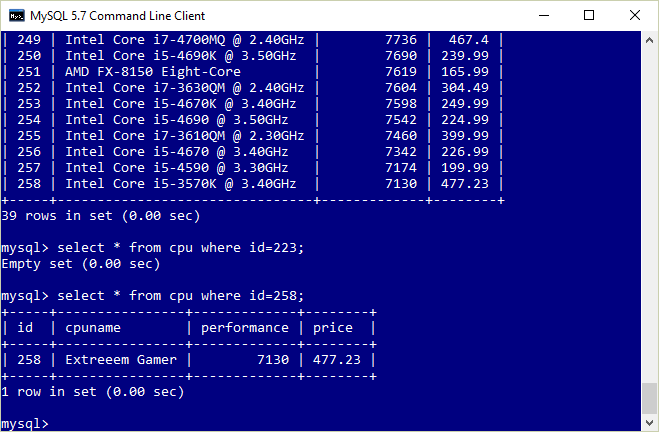
The user can edit the table values and the application will update the contents of the database. The following screenshot shows how this works on the Cpu name field.



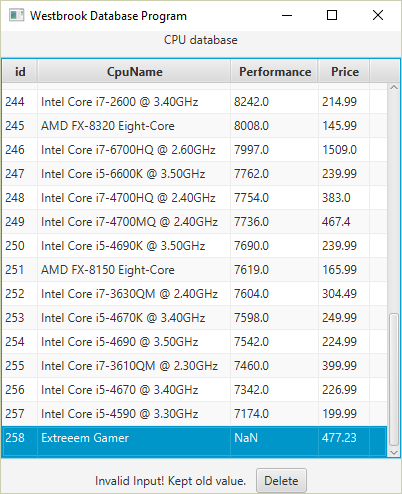
Then the user updates the name as shown:



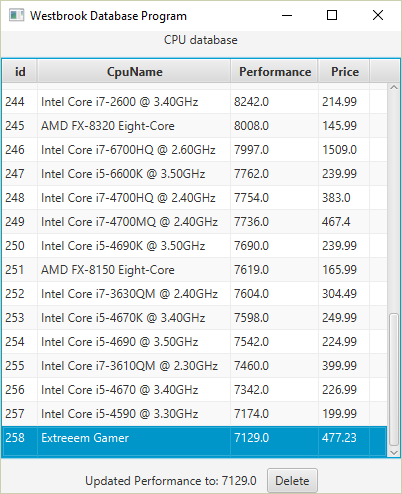
Then you can see that the database has been updated:



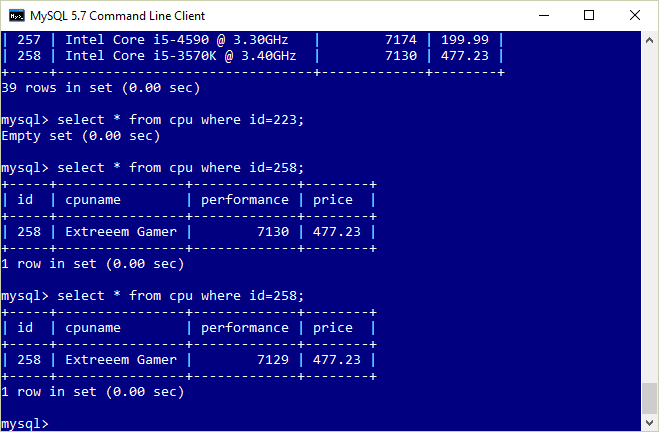
Similarly, the user can update the performance and price values but there is some validation going on. If the user enters a Double, everything is fine but if the user accidentally enters anything else besides numbers and a decimal point, the value is not changed in the database and the user is notified at the bottom of the window. See the following example:



See the following for a valid value and an appropriate database update.



And the proof:



This fulfills the requirements set out in the Project07 assignment. The attached jar file contains classes and source – all as directed in the assignment instructions except that it is not executable.