Assignment 4
Graphical User Interface (GUI) with JavaFx

Weight: 5% of final grade

Work with a partner (strongly recommended, but not mandatory).

Demo booking link:

https://docs.google.com/spreadsheets/d/13bcU9tTmNYe7OsWLjlR8Je6QO-ZCpt8Z_JGmtOTR6ss/edit?usp=sharing

The Problem

In this assignment you will use JavaFX to create a graphical user interface for the toy store software you wrote in Assignment 3. This means you will primarily be replacing the view components of your previous assignment with graphical interfaces. Some corresponding changes will be needed in the controller components too (the severity of these changes will depend partly on your attention to modularity and object-oriented design during assignment 3 😉)

Requirements

Your project must have the following features and specifications:

Project Directory Structure

Follow this directory structure:

- bin/ (compiled java files)
- src/ (Java source files)
    - application/ (classes housing the "main" program)
    - controller/ (classes used to coordinate the "model"s and "view"s)
    - model/ (classes related to data objects, like toys)
    - view/ (classes related to user interface)
    - exceptions/ (custom exception classes)
- lib/ any 3rd-party libraries. This folder may be empty
- res/ any resources or data files
- test/ unit tests

Exceptions

The program should still throw the custom exceptions you implemented in Assignment 3:

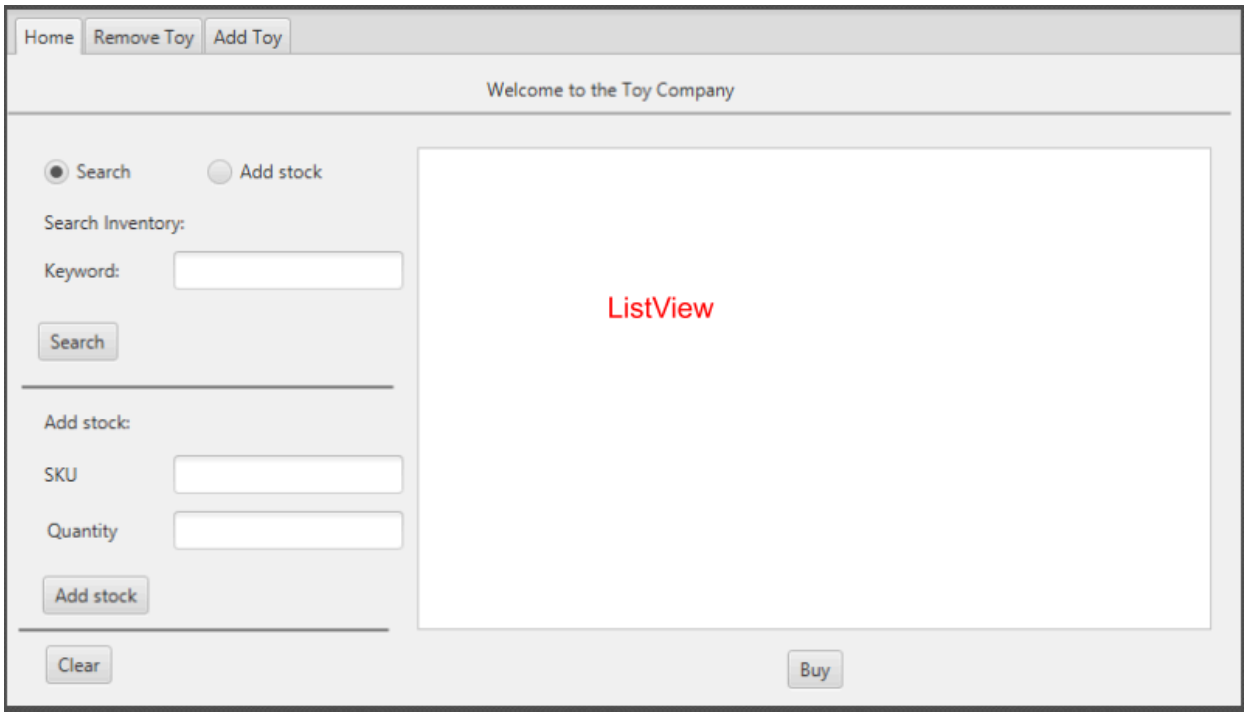
- The user enters invalid values when entering a new product
- The user attempts to purchase a product that is out of stock
- The user attempts to stock a product that is not physically stocked (e.g. a video game)

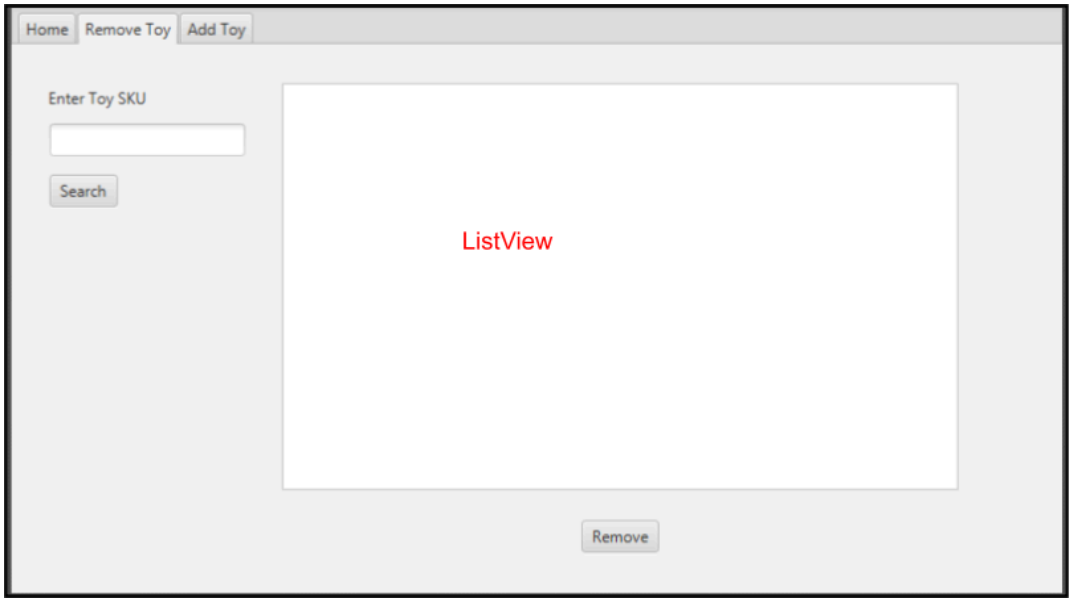
Testing

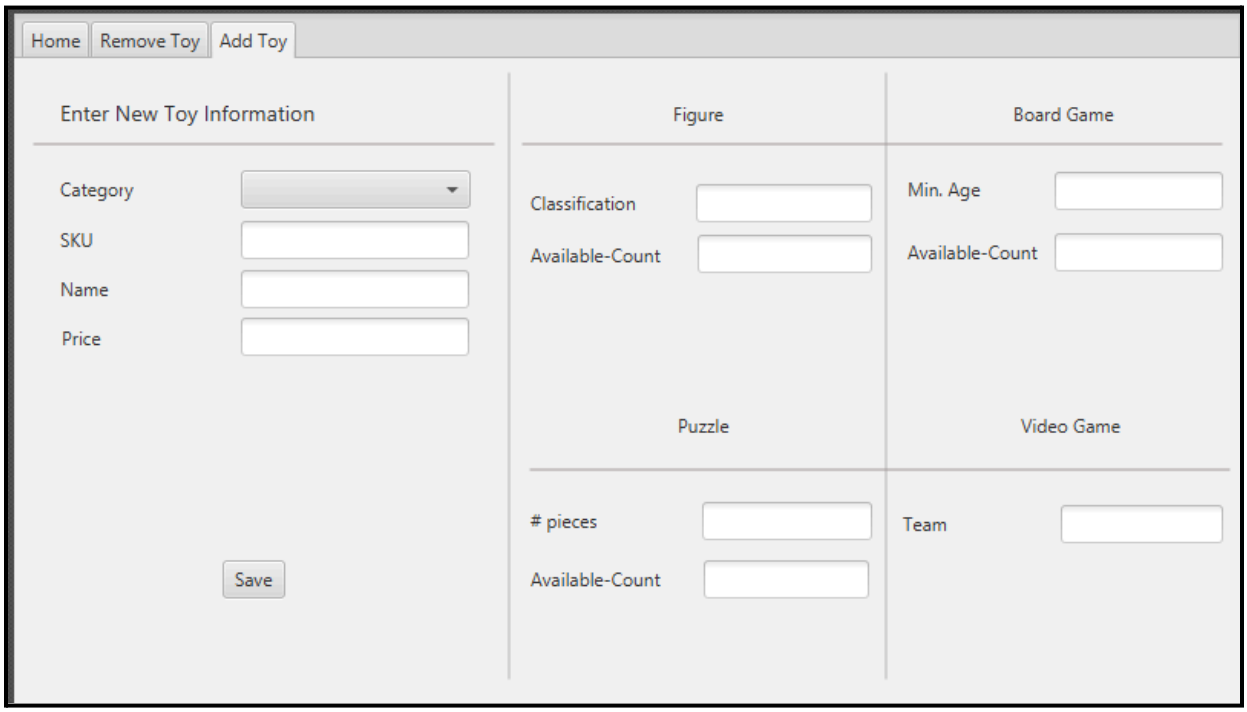
I expect to see **further** JUnit tests for your project when compared to the reasonable tests implemented for Assignment 3. Isolate some key functions and write tests for them that cover **all** possible situations. Go beyond testing getters/setters and test key operations that your program performs.

GUI components

The exact layout is up to you, but at a minimum your GUI must have the components shown in these sample screenshots that support the attributes described in the previous assignment..







Team work

- Before starting, understand the requirements and meet with your project partner to divide the tasks equally..
- Both members of a team must be knowledgeable about the whole project. This must be evident during the assignment demo. Your instructor may ask questions or make requests of either team member.

Resources

- Video on how to apply MVC on a FX project & ListView here. Main content starts around 7:00. The instructions before that relate to JavaFx project setup, but you may prefer the instructions in the course slides.

Submission instructions

- Push all git commits before the due date. The contribution of each team member must be visible in Git history. Paste the link to your github repo into the D2L assignment to finalize your submission.
- You must demo your project to your instructor during a demonstration to be scheduled during tutorial/lab time. This demo is mandatory. No demo = a grade of 0
    - Both members of a team must be knowledgeable about the whole project. This must be evident during the assignment demo. Your instructor may ask questions or make requests of either team member.

Grading

The following items are prerequisite to having your project graded. Failure to satisfy any of these points could result in a grade of zero.

- Your code runs to completion. This doesn't necessarily mean it works perfectly, but programs that crash immediately when run may not be graded.
- You must participate in a code demo to be scheduled by your instructor during lab/tutorial time.
- Only techniques that have been taught in class are allowed - if in doubt, ask your instructor.
- Code must compile and run. Include a jar file in your finished project.
- Basic project structure follows the folder structure outlined above

Projects that satisfy the above criteria will be graded according to the following rubric:

Points will be awarded in each of the following areas:

- [6 points] Bug fixes as agreed upon with the instructor **[Lab 9]**
    a. the program works without error, and produces correct outputs)
- [6 points] GUI implementation **(bonus points)**
    a. the program works without error, and produces correct outputs)
- [6 points] Object-oriented programming
    a. Problem decomposed into classes and methods appropriately, appropriate abstraction hierarchy, efficient code, etc.)
- [6 points] JavaDoc & Unit Tests
- [6 points] Your program was demoed using a runnable JAR file in the repo