CPSC 219: Introduction to Computer Science for Multidisciplinary Studies II

Assignment 3: JavaFX, GUI, Event-Driven Design, SceneBuilder, Object-Oriented Java

Weight: 10%

Collaboration

Discussing the assignment requirements with others is a reasonable thing to do and an excellent way to learn. However, the work you hand in must ultimately be your work. This is essential for you to benefit from the learning experience and for the instructors and TAs to grade you fairly. Handing in work that is not your original work but is represented as such is plagiarism and academic misconduct. Penalties for academic misconduct are outlined in the university calendar.

Here are some tips to avoid plagiarism in your programming assignments.

1. Cite all sources of code you hand in that are not your original work. You can put the citations into comments in your program. For example, if you find and use code found on a website, include a comment that says, for example:

   ```
   # The following code is from
   https://www.quackit.com/python/tutorial/python_hello_world.cfm.
   ```

   Use the complete URL so that the marker can check the source.

2. A tool like chat-GPT can be used to improve small code blocks. For example, five lines of code. If you get help from code assistance like Chat-GPT, you should comment above the block of code you requested assistance on debugging or improving and cite the tool used to get that suggestion. Using a tool like chat-GPT to write the majority of your assignment requirements will be treated as plagiarism if found without citation, and with citation, it will be treated as 0 for the component the student did not complete. Code improvement of short length will get credit if commented/cited properly.
3. Citing sources avoids accusations of plagiarism and penalties for academic misconduct. **However, you may still get a low grade if you submit code not primarily developed by yourself. Cited material should never be used to complete core assignment specifications. Before submitting, you can and should verify any code you are concerned about with your instructor/TA.**
4. Discuss and share ideas with other programmers as much as you like, but make sure that when you write your code, it is your own. A good rule of thumb is to wait 20 minutes after talking with somebody before writing your code. If you exchange code with another student, write code while discussing it with a fellow student, or copy code from another person's screen, this code is not yours.
5. **Collaborative coding is strictly prohibited**. **Your assignment submission must be strictly your code**. Discussing anything beyond assignment requirements and ideas is a strictly forbidden form of collaboration. This includes sharing code, discussing the code itself, or modelling code after another student's algorithm. **You can not use (even with citation) another student's code.**
6. Making your code available, even passively, for others to copy or potentially copy is also plagiarism.

7. We will look for plagiarism in all code submissions, possibly using automated software designed for the task. For example, see Measures of Software Similarity (MOSS - https://theory.stanford.edu/~aiken/moss/).
8. Remember, if you are having trouble with an assignment, it is always better to go to your TA and/or instructor for help rather than plagiarizing. A common penalty is an F on a plagiarized assignment.

Late Policy

Assignments will be accepted up until 48 hours late with a penalty. From $0 < hours \leq 24$ will result in 10% penalty. From $24 < hours \leq 48$ will result in a 20% penalty.

Goal

Writing an Event-Driven Object-Oriented GUI program in **Java** using **JavaFX** and **SceneBuilder 2.0**. Continue to use **Git** version control properly to store this project.

Technology

Java 20, Git, JavaFX, SceneBuilder 2.0

Submission Instructions

You must submit your assignment electronically using **Gitlab** and **D2L**. Use the Assignment 2 dropbox in **D2L** for a final codebase electronic submission. You will also share a link to your **GitLab** codebase with your TA in that D2L submission. In **D2L**, you can submit multiple times over the top of a previous submission. Do not wait until the last minute to attempt to submit. You are responsible if you attempt this and time runs out. Your assignment must be completed in **Java** (not Kotlin or others) and be executable with **Java version 20.** You must use **Gitlab** hosted at **https://csgit.ucalgary.ca/** (not GitHub or another Git host). You must use **JUnit 5** and not other unit-testing libraries.

Description

You are going to complete a Map Making program from your MonstersVsHeroes simulation game. That game was relatively simple to that of modern fantasy-world video games and had have no graphical portions or any real degree of user input. You will be creating a graphical input/output tool to edit the maps that the MonstersVsHeroes game was able to load and simulate.

The goal of this assignment is to expose students to an Event-Driven Object-Oriented (OO) Graphical User Interface (GUI) program. You will be provided with an existing framework of code which is already in an OO format. **This is the same framework from Assignment 2.** This means concepts of the game have been broken off from one single procedural file and stored (encapsulated) in separate files according to their purpose. **These files will be fully complete, except for file reading which you should re-use your code from A2.**

There will be a variety of OO concepts visible throughout the code. Code will be stored in packages. Some code will be built as procedural helper classes like the incomplete **Reader**. Other code will be Enum types like **Direction**/**WeaponType**/**Symbol** and internal **State** enumerations. Some code will be a regular object-type like **World**. Some code will be abstract like **Entity**, and other code will extend that abstract code like **Wall**/**Hero**/**Monster**. This variety of OO code will help students understand a variety of OO concepts including encapsulation, polymorphism, enumerations, abstract, and static.

You will add a new package at mvh.app with a Main/MainController/Main.fxml to create a **GUI**. This GUI will be designed to allow the user to create/save/load/edit a world file.

Do not change the existing portions of the code. Create the new files only where indicated, and add functionality as specifically requested. SHOULD NOT IMPORT ANY OTHER LIBRARIES BEYOND java.io/java.util and JavaFX libraries FOR THIS ASSIGNMENT and you will get not credit for code copied from other places (cited code will get a grade of 0 as you are required to complete the required functions yourself).

Git Requirement: For this assignment we will have the requirement that you upload your code to the university csgit.ucalgary.ca hosting site as a **private** repository shared with your TA (in addition to submitting it via D2L dropbox). The minimum expectation for Git usage is that when you submit your code that the TA can go and view it in Gitlab and see that you've made multiple commits as you edited it before the submission deadline to D2L. To use csgit.ucalgary.ca you will need a functional **UofC IT account** username/password.

You will be required to submit your regular Main.java/MainController.java/Main.fxml source files.

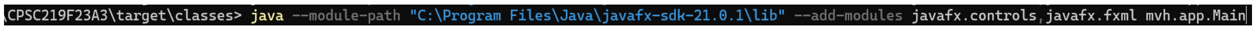
To Run My GUI

It is sufficient to run your GUI through your IDE as a JavaFX program.

However, it is also possible to run it with something like the following if you only use the base JavaFX components.

java --module-path "C:\Program Files\Java\javafx-sdk-21.0.1\lib" --add-modules javafx.controls,javafx.fxml mvh.app.Main

I executed this in ***yourprojectname*****\target\classes** folder after building my project.



```
\CPSC219F23A3\target\classes> java --module-path "C:\Program Files\Java\javafx-sdk-21.0.1\lib" --add-modules javafx.controls,javafx.fxml mvh.app.Main
```

Your **--module-path** will be different than mine. I downloaded the latest JavaFX sdk from https://gluonhq.com/products/javafx/ for Windows (my OS) and installed it. Then I used the location of installation **"C:\Program Files\Java\javafx-sdk-21.0.1\lib"** to run my program.

You should include a **readme.md** instruction with your code that includes this run execution description so that you remember in the future how to run your project code.

You should also create a packaged **CPSC219F23A3.jar** file that you will submit. Jar files are an artifact that you can create with your IDE. They are zip files that store your project .class files and other parts of program inside of it. I can run a jar file with my compiled project inside with the following.

java --module-path "C:\Program Files\Java\javafx-sdk-21.0.1\lib" --add-modules javafx.controls,javafx.fxml -jar CPSC219F23A3.jar

Example world.txt file

```
3
3
0,0,MONSTER,M,10,S
0,1
0,2
1,0
1,1
1,2
2,0
2,1
2,2,HERO,H,10,3,1
```

3 rows x 3 columns grid. First entry is rows, second is columns.

Monster is at (0,0) location with symbol M, health 10, and weapon type of (S)word. Other WeaponType options are (C)lub and (A)xe. Sword has attack strength of 4, Axe 3, and Club 2.(All Monsters have a static armour strength of 2 that is not entered via the file.)

Hero is at (2,2) location with symbol H, health 10, attack strength 3, armour strength 1.

When loaded in the game, this would look like the following (note, this is a 3x3 world in the file, but we are drawing it for viewing with external walls outside the valid array indices as #):

```
#####
#M..#
#...#
#..H#
#####
```

Where # are Walls. M is a Monster, and H is a Hero.

GUI Requirements

Title

1. Name of program and version

Menu Bar

1. Menu Item (File)
    a. Load file
        i. Load file from FileChooser (can re-use Reader code from A2)
    b. Save file
        i. Save file to FileChooser (will need to write this code)
    c. Quit
2. Menu Item (Help)
    a. About
        i. Alert with info about you/program

Create World

1. Row/Column input, button to create

Add Hero/Monster

1. Symbol
2. Health
3. WeaponType or Weapon/Armor

Delete Entity

1. Replace with null in world

View World

1. See world you are editing and update after each change

View details of Entity

1. Give location and get info about entity at row/column

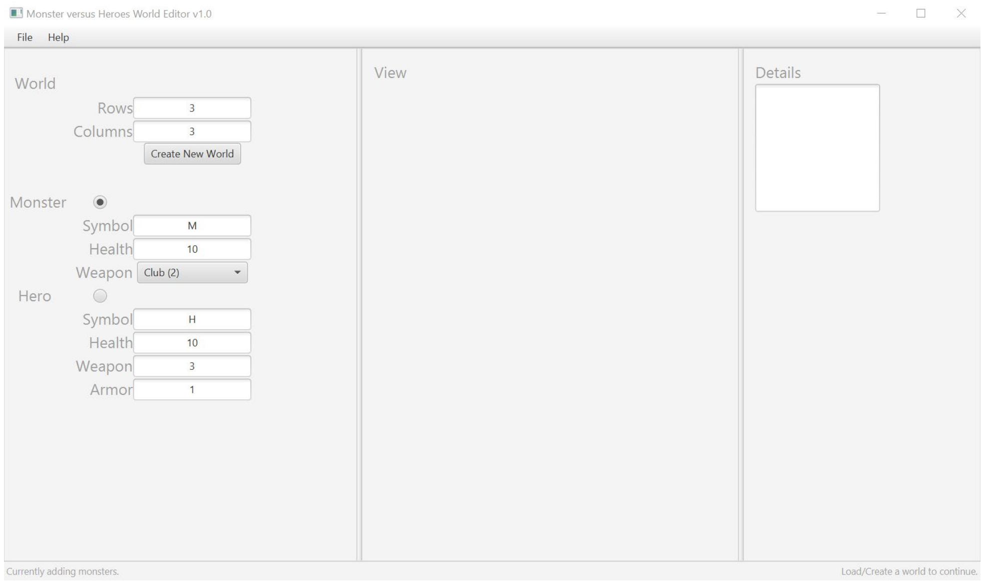
Status

1. At bottom of window (will show success/failure messages)

There is no strict requirement on how your program looks. Or what buttons/labels/layouts you choose.

Here’s an example of what I may have created to complete this assignment. Your solution does not have to look like mine.

Basic view with 3 panels and menu bar.



Rows 3
Columns 3

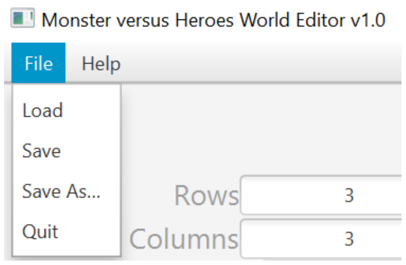
Symbol M
Health 10
Weapon Club (2)

Symbol H
Health 10
Weapon 3
Armor 1

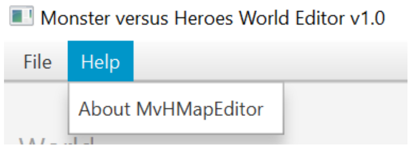
Currently adding monsters.

Load/Create a world to continue.

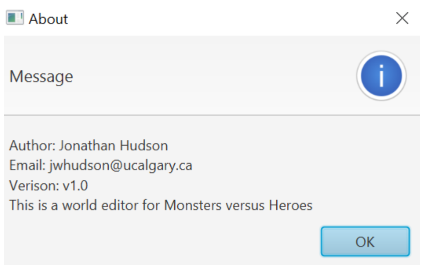
The menu options. I added both a ‘Save’ at world.txt and a ‘Save As’.



Launch the about



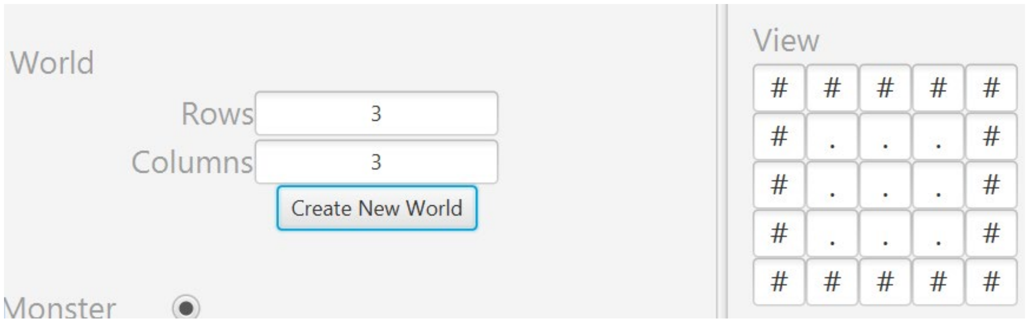
The info



Message

Author: Jonathan Hudson
Email: jwhudson@ucalgary.ca
Verison: v1.0
This is a world editor for Monsters versus Heroes

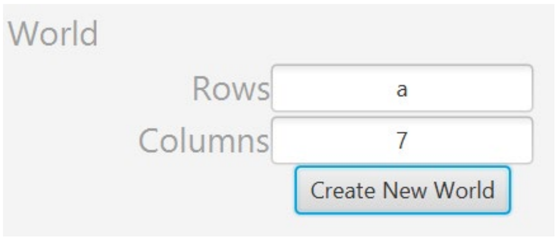
What does a new world look like. (Note, my view of world is more complicated than I expect students to do. Student can just write the worldString to text output instead of this more complex Grid setup I have designed.)



Rows 3
Columns 3

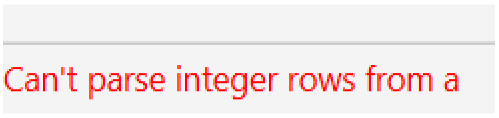
```
# # # # #
# . . . #
# . . . #
# . . . #
# # # # #
```

What if something goes wrong!



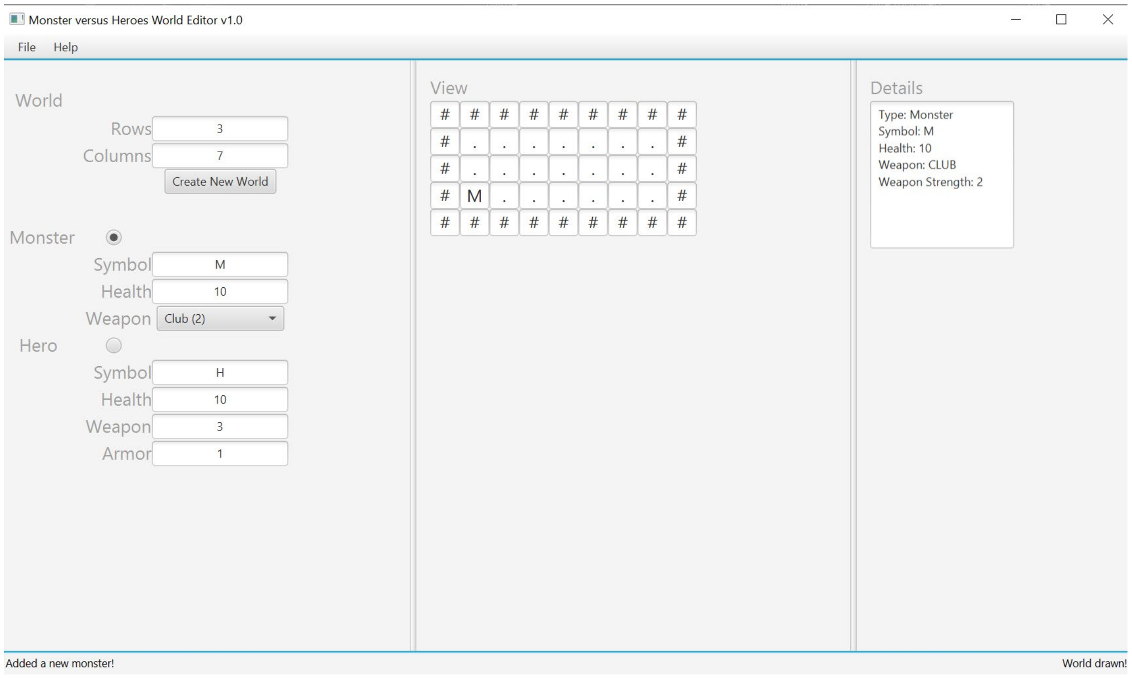
Rows a
Columns 7

I have a left/right status outputs. I use left for the user actions and right for internal issues like drawing world or issue getting details about entity.



Can't parse integer rows from a

Added a monster and seeing the info about it. (As before, my world view is more complicate than expected for students. I add monsters by clicking primary button and remove when clicking secondary button on mouse. I view monsters on ‘mouse over’ a grid spot. You can add by user typed row/column location, and view in the same way. No need to use ‘mouse click/over’ methods on a Grid like I have.)



Rows: 3
Columns: 7

Monster [✓]
Symbol: M
Health: 10
Weapon: Club (2)

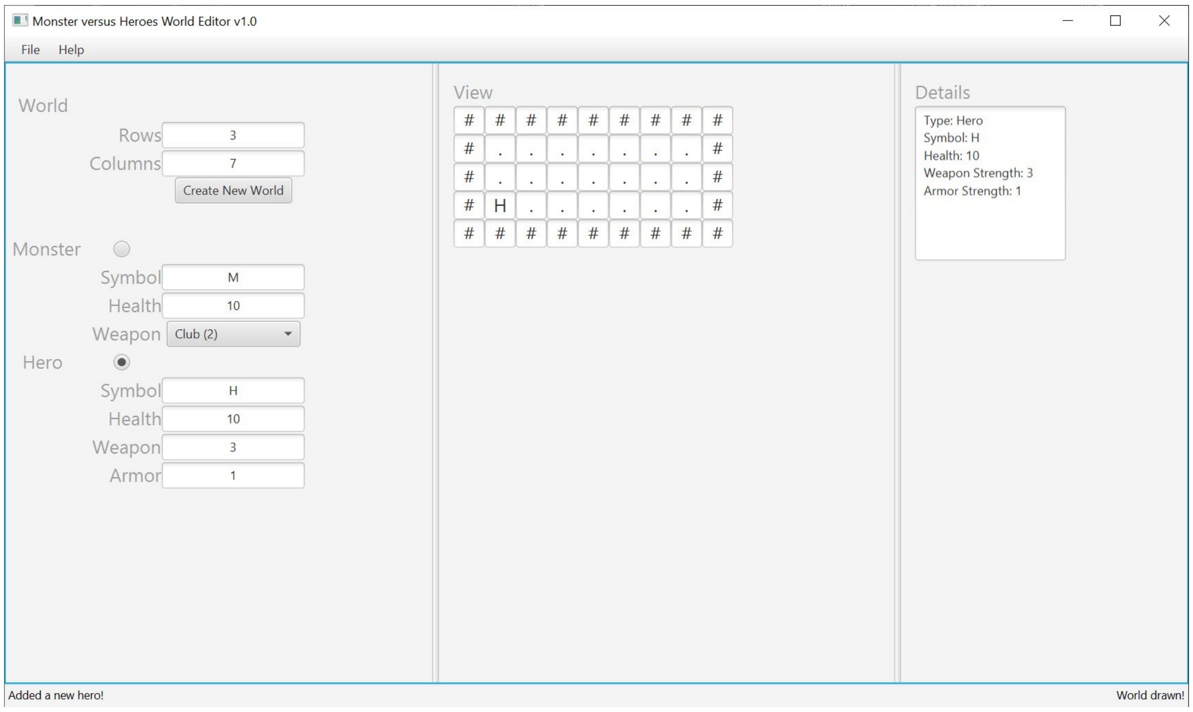
Hero
Symbol: H
Health: 10
Weapon: 3
Armor: 1

```
# # # # # # # # #
# . . . . . . . #
# . . . . . . . #
# M . . . . . . #
# # # # # # # # #
```

Type: Monster
Symbol: M
Health: 10
Weapon: CLUB
Weapon Strength: 2

Added a new monster! World drawn!

Added a hero and seeing the info about it.



Rows: 3
Columns: 7

Monster
Symbol: M
Health: 10
Weapon: Club (2)

Hero [✓]
Symbol: H
Health: 10
Weapon: 3
Armor: 1

```
# # # # # # # # #
# . . . . . . . #
# . . . . . . . #
# H . . . . . . #
# # # # # # # # #
```

Type: Hero
Symbol: H
Health: 10
Weapon Strength: 3
Armor Strength: 1

Added a new hero! World drawn!

Bonus

Change this map editor into a game simulator. Users should be able to

1. Start simulation (try to run to inactive)
2. Stop simulation (if simulation is started this will stop it)
3. Advance 1 simulation step (if active)
4. Edit world (both active/inactive)
5. See a counter of simulation steps updates as simulation progresses
6. Still able to save/load a partially simulated world

The world should update as it is being played on the screen. Use a time delay on the continuous play option so that users are able to view updates in a reasonable way that they can then stop it when they want to.

Additional Specification

- You must comment your code with **Javadoc** comments.
- **Use in-line comments** to indicate blocks of code and describe decisions or complex expressions.
- **Do not use inline conditionals**
- Put your **name, date, and tutorial** into the comments for the **Javadoc** of the class for your TAs to identify your work.
- **Do not rename the provided files.** You must you exactly the request filenames.
- **You should not import ANY libraries (outside java.io/java.util/JavaFX) to complete the regular assignment. Using these could result in grade of 0 for that portion of assignment. Citing code from internet to complete a function will also result in 0.**
- Use constants appropriately. Your TA may note one or two magic numbers as a comment, but more will result in lost marks.
- Do not change provided code without discussion with instructor. If there is a bug, or something is broken, the instructor should be informed to fix this issue.
- **You should perform error for user input in the GUI you create and output messages to the status area.** The provided code should be designed to assume the rest of program is only inputting valid inputs.

Grading

The total grade is out of 50. Bonus marks can reach up to 55 marks if completed.

GUI Files (out of 40) [TAs will grade partially by running your code and interacting with it]
Title 1
Menu Bar Parts 1
Menu Bar Load Operation 4
Menu Bar Save Operation 4
Menu Bar Quit Operation 1
Menu Bar About Operation 2
Create World 3
Add Hero/Monster 5

Add Hero/Monster 5
Delete Entity 2
View World 4
View details of Entity 2
Status 2
Readme.md 2
.jar file 2
Gitlab Usage (out of 5)

Gitlab account exists, private project exists, at least 1 commit, small commits, regular commits

Style/Commenting (out of 5)

Name/Date/Tutorial, Functions commented, Javadoc, Inline commenting, doesn't use inline conditionals, limited magic numbers, don't change function names, don't change filenames, etc.

BONUS MARKS (out of 5)

- Start simulation (try to run to inactive) 1
- Stop simulation (if simulation is started this will stop it) 1
- Advance 1 simulation step (if active) 1
- Edit world (both active/inactive) 1
- See a counter of simulation steps updates as simulation progresses 1

Invite your TA to your private!!!! Csgit.ucalgary.ca project for the assignment

Submit the following using the Assignment 3 Dropbox in D2L

1. Main.java/MainController.java/Main.fxml (all should be mvh.app as their package)
    a. Include Reader.java (can be your A2 Reader, or what you made for A3 if you didn't finish this in A2, you might also just have this code in your MainController)
    b. Include a Writer.java (this is new from A2, but you may have code in your MainController instead)
    c. Include World.java if you edited it as well
2. readme.md (explains how to run program and jar file base on user having Open JavaFX)
3. CPSC219F23A3.jar
4. Name of your private csgit.ucalgary.ca repository that you've added your TA to as at least a Developer