CSCI 2600 — Principles of Software

Optional Extra Credit Homework 8: RPI Campus Paths GUI

Due: Wednesday, Apr. 24, 2024, 11:59:59 pm

Submission Instructions

- This assignment uses the same repository as Homework assignments 4, 5, 6, and 7, so when you are ready to start working on Homework 8, pull Homework 8 files from the repository by right-clicking on your Homework 4, 5, 6 and 7 project in Eclipse and selecting `Team` → `Pull...` Make sure that `When pulling` is set to `Merge`, then click `Finish`.
- Be sure to commit and push the files to Submitty. Follow the directions in the version control handout for adding and committing files.
- Be sure to add any additional files to your repo using Team/Add to Index.
- **Important:** You must press the **Grade My Repository** button, or your answers will not be graded.

IMPORTANT NOTES:

You should have package `hw8` with the usual directory structure. Write your code under `src/main/java/hw8` and your tests under `src/test/java/hw8` (shows as `hw8` under `src/test/java` in Package Explorer).

Introduction

In Homework 7, you wrote a program that found shortest routes between buildings on the RPI Campus. Now we are asking for a graphical user interface (GUI) that visually draws routes.

You will build your GUI using Java's JavaFX library. In completing this assignment, you will get practice using JavaFX, event-driven programming, and the MVC (Observer) design pattern.

Important note: you are not allowed to use Swing, AWT, or any library or framework other than JavaFX and the standard Java library.

You are expected to fix any bugs from Homework 7 that affect the correctness or performance of your application in Homework 8. Furthermore, your Homework 8 should use the model you created in Homework 7. This may require that you modify your Homework 7 code, that is OK, but be sure that Homework 7 continues to pass all tests!

There is no auto-grading on this assignment. However, you still must submit through Submitty.

GUI Requirements

You will write a GUI and a main class to launch it named `RPICampusPathsMain.java`. This assignment is deliberately open-ended: the exact appearance and functionality of your GUI are up to you. The only requirements are documented below.

For the most part, we are not grading on aesthetics: it doesn't matter whether your GUI looks pretty as long as it implements the required features. Nevertheless, a design which is genuinely confusing or hard to use (at our discretion) may not receive full credit. For example, we will deduct points if we can't easily figure out how to select the two buildings, if it's hard to see the selected path, or if we can only see the whole GUI on a 27-inch screen. In addition, your program should be generally responsive: for instance, the GUI should not take an unusually long time to find and display paths.

Your GUI is a new View and Controller for your Campus Paths application. Ideally, you should not have to make any changes to your Homework 7 model classes — they already implement all the model functionality that you need. If you have to make any small changes (for instance, if your design in Homework 7 was poor and some model methods were too closely tied to your text view), then you may do so. As always, all tests from previous homework assignments must continue to pass, so you may also need to change your Homework 7 View and Controller in that case. In file `answers/hw8_model-changes.pdf`, list any changes you made to the model. For each, write a 1-2 sentence explanation of why the change was necessary and what you could have done differently on Homework 7 to create a more general and reusable Model. If you made no changes, write "None" for this section.

Window size

At startup, your GUI must fit and be usable on a screen with resolution 1024 x 768 and above. Most computers provide a way to change the screen resolution, which you can use for testing.

Required features

Your GUI must provide the following features:

- At startup, load the map data from `data/RPI_map_data_Edges.csv` and `data/RPI_map_data_Nodes.csv` which you should already have in your repository from Homework 7. This should be in your model, not your view. There is no need to duplicate files; load them directly from the `data/` directory.
- Display the map of RPI campus. You may remove the part at the bottom with the building names if you need.
  Important note: download the map from the course Web site and save it into your data/ directory: data/RPI_campus_map_2010_extra_nodes_edges.png. DO NOT commit the map as the large file may break the limit on repo size. When testing, we will copy the map into your data/ directory under the name RPI_campus_map_2010_extra_nodes_edges.png.
- Allow the user to select two buildings for finding a route. You need to allow the user to

select endpoints of a path by clicking with the mouse on the map. In addition, you may also implement some other approach, like selection using menus or dropdown lists.

- Mark or highlight the selected buildings and/or path endpoints on the map.
- Draw the shortest route between the selected buildings on the map. The map should automatically be zoomed in or out when a route is drawn, so that the route is almost as large as possible while still fitting in the window.
- As the window is resized, make the map shrink or grow to fit the window.
- Maintain the proportions of the map so that it zooms in on a route without becoming distorted.
- Place the map in a `ScrollPane` so it can be displayed full-size. When displaying a route or buildings, jump to that spot on the map to resize if needed. (Hint: you probably need to override `getPrefViewportHeight()` and `getPrefViewportWidth()` in your "canvas" class for scrolling to work.)
- Allow the user to drag the map with the mouse to change the portion that is shown.
- Add zoom buttons, possibly with a way to recenter the image for zooming if a hand is not available to drag it (e.g., mouse double click).
- Allow the user to reset the GUI by clicking a reset button. This button should clear all markings on the map and all other controls (such as building selectors), setting the GUI back to its initial state.
- Operate robustly. No matter what the user does, your program should never allow an exception message to bubble up to the console window, and your GUI should never crash, freeze, display rendering artifacts, or reach a buggy/invalid state.

JavaFX scene elements and GUI builders

Use only components from the JavaFX library for this assignment.

Some IDEs, such as NetBeans, will let you specify the appearance and behavior of your GUI and automatically generate the code for you. JavaFX also implements FXML, a markup language from which GUI code can be generated. You may not use these tools; you must write your GUI from scratch in Java.

1 Launching your GUI

We will launch your GUI from Run As → Java Application.

List all features you implemented in `answers/hw8_features.pdf`. You must commit `answers/hw8_features.pdf` file to get credit for any features of your solution.

Writing automated tests for GUI is difficult and usually involves special frameworks that are beyond the scope of this course. For this reason, you are not required to write unit tests. We will test your solution by running your main program.

Reflection [0.5 points]

Please answer the following questions in a file named `hw8_reflection.pdf` in your `answers/` directory. Answer briefly, but in enough detail to help you improve your own practice via introspection and to enable the course staff to improve Principles of Software in the future.

(1) In retrospect, what could you have done better to reduce the time you spent solving this assignment?

(2) What could the Principles of Software staff have done better to improve your learning experience in this assignment?

(3) What do you know now that you wish you had known before beginning the assignment?

We will be awarding up to 1 extra credit point (at the discretion of the grader) for particularly insightful, constructive, and helpful reflection statements.

Collaboration[0.5 points]

Please answer the following questions in a file named `hw8_collaboration.pdf` in your `answers/` directory.

The standard integrity policy applies to this assignment.

State whether you collaborated with other students. If you did collaborate with other students, state their names and a brief description of how you collaborated.

Grade Breakdown

- Model Changes: 5 pts.
- Basic Functionality of GUI Application: 11 pts.
- Features of GUI Application: 3 pts. per required feature
- Collaboration and reflection: 1 pt.

Hints

General GUI Advice

If you have never used JavaFX, it is well worth your time to study some tutorials, read the example code, and generally get comfortable with GUI programming before diving into the assignment.

Abstraction functions, representation invariants, and checkRep() are not required for GUI classes because they generally do not represent ADTs.

User testing is a great way to verify that your interface is as easy to use as you think it is. Show your GUI to your friend/roommate/family member. Can they figure out how to use it without directions from you?

As usual, remember to follow good methodology and class decomposition among other best practices for style.

Programming With JavaFX

Oracle's JavaFX tutorials are a useful resource. Also remember to use the Java API and JavaFX documentation, to see what classes and methods are available and how to use them.

What to Turn In

You should commit and push the following files to Submitty. Don't forget to click "Grade My Repository" button on Submitty!

- `src/main/java/hw8/RPICampusPathsMain.java`
- `src/main/java/hw8/*.java` *[your GUI classes]*
- `answers/hw8_model-changes.pdf` *[list of changes to HW7. The file may simply contain "None".]*
- `answers/hw8_features.pdf` *[list of all features you implemented.]*
- `answers/hw8_reflection.pdf`
- `answers/hw8_collaboration.pdf`

Errata

Check the Submitty Discussion Forum for possible errata or other relevant information.

Q & A

None yet.

Parts of this homework were copied from the University of Washington Software Design and Implementation class by Michael Ernst.