CMPT 381 Assignment 2: UI Design and Development + MVC

Due: Friday, Oct. 15, 11:59pm

Overview

In this assignment you will design and build a user interface to demonstrate your skills with both the iterative design process (including task descriptions, sketch-based prototypes, screen design, and task walkthroughs) and JavaFX development (including widgets, events, layout, and Model-View-Controller). Part 1 covers early design activities, Part 2 covers implementation in JavaFX and MVC setup, and Part 3 covers MVC implementation and completion of the system.

Your job is to design and build a GUI for a time-tracker application (called TTrack) that lets users record the time they spend on a set of projects, create new projects to track, and see summaries of their time records. You will determine the primary tasks for this system, design a UI to support those tasks, and develop an MVC system based on your design.

Part 1: TTrack Design

Use the supplied persona below to provide general guidance on the important elements in the problem domain and criteria that you can use to design for a specific user population.

Persona for 381 A2 Design Problem

Name: Alex

Demographics and Experience: Alex is 32 and works as an independent software engineer in a large city. She owns a Labrador retriever, drives a Nissan Leaf, has season tickets to the opera, and plays competitive disc golf. Alex is very experienced with desktop and mobile systems, and is an expert in many different applications. Alex typically works on a PC with a large monitor, and always has numerous windows open so that managing screen space is a constant issue.

Goals and Needs: Alex works on several different software projects at a time (6-10), and new projects can start at any time. She needs a way to keep track of when she has put in time on each of her contracts (as well as how much). She bills each contract monthly, based on the amount of time worked on that project. Although Alex does not bill by the second, different work periods are added up for each billing period, so she needs a fine-grained record of her work time.

Motivations: Alex works long hours, and she puts a high value on efficiency – as a result, she is not very patient when systems make her wait unnecessarily, force her to do extra work, or cause her to make errors.

1A Task Descriptions

Identify a set of candidate tasks for the TTrack interface, using the persona description above as well as your own understanding of what is required in a time-tracker system. Using the ideas of importance and frequency, choose the top three candidates and write these up as task descriptions. Refer to the 381 lecture notes for information on what goes into a task description, and include the following headings in your description:

- Task Name
- Persona
- Importance and frequency (including justification for your ratings of these two factors)
- Description (include specific data that will be used in the task)
- Constraints and goals relevant to the task

Write the three task descriptions (maximum ½ page each, single-spaced, 12-point font, 2.5cm margins) and save the document in PDF format.

The product of your work in part 1A will be three written task descriptions in a PDF document.

1B Interface Sketches

Based on the information in the persona and your task descriptions, create sketches for different interface designs that will support each of your three task descriptions. You should create a wide variety of sketches for each task, but you will only hand in three sketches for each (9 sketches in total). The sketches that you hand in should be complete enough that you can carry out task walkthroughs, and should cover a range of approaches (do not, for example, hand in three variations on a single idea for any task). Refer to the 381 lecture notes and readings as a guide to producing your sketches, and remember that your ideas don't all have to be good ones (and, your artistic ability will not be evaluated).

Take photos of your 9 sketches, and name the files with the number of the task and a number for the sketch (e.g., task1-sketch1.png). If any sketches have multiple parts, label the pictures to clearly indicate what is shown (e.g., write the label on paper and put it on the sketch before you take the picture), and ensure that the filenames record the relationship (e.g., task1-sketch1a.png, task1-sketch1b.png, etc.).

The product of your work in part 1B will be digital photos of your 9 sketches, with appropriate filenames

1C Task-Based Walkthroughs

For each of your 9 sketches, carry out a task-based walkthrough using the corresponding task description. Make notes about what worked well and what caused problems. Create a report that shows a picture of each sketch, a point-form summary of the walkthrough results, and a final assessment of the overall strengths and weaknesses of each design. Organize the document as a table, with the sketch shown at left and the walkthrough results and assessment on the right. The report for each design should take about ½ page (so, no more than 5 pages for your 9 designs).

The product of your work in part 1C will be a PDF document with the table described above

Part 2: View Redesign & Development, and MVC Setup

In this part of the assignment you will revise your designs based on your task walkthroughs (and based on what you will be able to build in JavaFX) and set up your system's MVC architecture.

2A View Redesign & Development

Using the results of your task walkthroughs, and considering what is possible in JavaFX, do a final redesign of your best designs from part 1 to produce a version that you will turn into a JavaFX UI. You will have three final sketches.

Build view classes in JavaFX, based on your final sketches, that support your chosen tasks. Follow the guidelines presented in 381 lectures for developing your views. Note that there is no requirement that you have three classes for your three tasks – the way you implement your sketches is up to you, as long as your JavaFX views clearly represent your sketches, and as long as they support your chosen tasks. However, any views that you create should be separate classes in JavaFX. Create a document (called sketches-to-fx.pdf) that shows, for each task, your final sketch for that task, a screenshot of the JavaFX UI that represents that sketch, and notes to explain any design changes that were necessary between the sketches and the JavaFX versions.

The product of your work in part 2A will be a PDF document that shows your final sketches, screenshots of your JavaFX views, and notes to explain design changes.

2B MVC Setup

In addition to your view classes, create classes for a model and controller. In your application class, write code to create and connect all elements of MVC, following guidelines shown in 381 lectures. Write only the controller and model methods needed for setup (including publish-subscribe communication).

The product of your work in part 2B will be MVC code (as part of the project handed in for Part 3 below)

Part 3: TTrack Implementation

In the third part of the assignment you will complete the implementation of your model and controller, handle view resizing, and add any other capabilities required to make the system operational.

3A Model and Controller

Write code that implements an appropriate model for your system, and that handles events for all needed user interactions.

- Your model will likely work with times and dates: the Java classes LocalDate and LocalTime (or the combined LocalDateTime) will be useful, and class java.time.Duration is useful for working with elapsed and total times
- Your model must use Java access control to restrict the model's public API as discussed in 381 lectures.
- Your system must implement the MVC communication approaches described in 381 lectures (e.g., your model must use publish-subscribe communication to notify views of changes)
- Your controller should handle all user input events, with a few exceptions (e.g., if you implement view switching, these events can be handled by the application class since it is this class that has access to the different Scene objects)
- You should maintain as much separation as possible between your M, V, and C components, as discussed in lectures.

The product of part 3A will be JavaFX code for your model and controller

3B Resizing and Final Polish

Add code to your system to ensure the following:

- If an application window is resized, the interface layout responds appropriately (within the constraints of the OS and windowing system)
- The different tasks supported by your system have a consistent look and interaction style
- The system allows a user to successfully carry out your three tasks successfully and meets the design goals that are part of the original persona and descriptions of the domain
- The user can navigate through the system to reach all of the views and carry out all of the intended tasks
- Note: you can make minor changes to your designs in this stage of the project – but do not switch to a completely different approach; your final application must show a comprehensible design progression
- Note: there will be user tasks that your application does not include; you do not have to produce a complete real-world application, just a prototype that meets the requirements described here

What to hand in (each student will hand in an assignment)

- Create a zip file of all of your assignment materials:
    - Directory "Part1": task description document, sketch images, and walkthrough report document
    - Directory "Part2": final design document
    - Directory "Part3": JavaFX/IDEA project zip (File → Export → Project to Zip file…).
- Add a readme.txt file to the zip that describes the organization of your handin and provides any comments to the markers about what to look at in your code (as always, systems for 381 should never require the marker to install external libraries). If parts of the system don't work, but you want us to look at the code for these parts, please explain in your readme.txt

Where to hand in

Hand in your zip file to the Assignment 2 link on the course Canvas site.

Evaluation

- Part 1A. Your task descriptions should follow the guidelines introduced in class, should be core to the requirements for the TTrack system and the supplied persona, and should follow the guidelines regarding length and formatting.
- Part 1B. Your 9 sketches should clearly relate to the task descriptions, should be clearly comprehensible (you do not have to show artistic talent but we need to be able to understand what is going on in the sketch), should be complete enough to allow task walkthroughs, and should cover a range of design possibilities.
- Part 1C. Your walkthrough report should follow the provided formatting guidelines, should provide clear and concise summaries of the walkthrough results, and should provide a meaningful assessment of the overall strengths and weaknesses of each design.
- Part 2A. Your final sketches should show progress from the initial versions based on the results of the task walkthroughs, and should maintain clear connections to your task descriptions. Your JavaFX screenshots should show a clear connection to the final sketches. Your JavaFX views should be implemented as classes, following the guidelines discussed in lectures (Note that your view code will be evaluated based on the project handed in for Part 3; you do not need to hand in code for Part 2A)
- Part 2B. Your MVC architecture should be developed using separate classes for each element, and should follow the communication guidelines presented in lectures. (Note that your MVC architecture will be evaluated based on the project code handed in for Part 3; you do not need to hand in code for Part 2B)
- Part 3A. Your model should provide a clear and comprehensible representation of the data that is to be stored for the application, should provide appropriate methods to add and manipulate the data, and should maintain appropriate access restrictions on methods and data structures. Your controller should handle user events using appropriate methods and should call appropriate methods of the model based on user events.
- Part 3B. Your finished system should handle window resizing appropriately, should allow a user (in particular, the given persona) to carry out the tasks that you identified at the start of the project, should support the main design goals and constraints that are set out in the persona and the task context, and should show a comprehensible design progression.

If parts of your system have only partial implementations (e.g., a feature does not work but has been partially developed), clearly indicate this in your readme.txt file. Code should be appropriately documented and tested (although documentation will not be explicitly marked). No late assignments will be allowed, and no extensions will be given, without medical reasons.