CS 4063/5063
Homework: Prototype E
Due Thursday 2022.04.21 at 11:00pm.

All homework assignments are individual efforts, and must be completed entirely on your own.

In this assignment you will continue to develop a movie collection application in JavaFX. You will learn how to write basic FMXL and FXCSS, and use it to implement an alternative version of the **summary** from DesignB. You will also (finally!) learn how to use the advanced styling features, graphical effects, and animation capabilities that make JavaFX fun to use for UI development. Specifically, you will implement the details of layout, styling, interaction, and animation of a new cover flow component to round out our development experience with the movie collection app.

Preparing for Implementation

The implementation work in this assignment is split into two parts. The first part is to write FXML and FXCSS to reimplement the **summary** design from the DesignB assignment. To do that, you will edit the three files in the `fxml` directory inside the `prototypee` package. Before you start, review the slides from class on FXML+FXCSS and spend some time familiarizing yourself with the guided tour and reference materials found at the following link:

https://docs.oracle.com/javase/8/javase-clienttechnologies.htm
click on *Build UI with FXML* for a tour of FXML capabilities
click on *Introduction to FXML* for a reference overview of FXML features
click on *CSS Reference Guide* for a reference overview of FXCSS features

The second part is to implement your **coverflow** design from the DesignE assignment. To do that, you will complete the two new Java files included in the `flow` directory of the `prototypee` package. Before you dive in, spend some time browsing the APIs of `javafx.scene.*` in which `*` = {`Animation`, `Effect`, `Paint`, `Shape`, `Text`, `Transform`}. You're likely to draw especially heavily from the `Paint`, `Shape`, and `Text` APIs to reproduce your **coverflow** design.

Implementing the Prototype

Start by putting a copy of your `DesignE.bmpr` file in the `Results` directory. *(We need your design file for comparison with your prototype UI. You can create a design file to include now even if you didn't finish the DesignE assignment.)*

The `prototypee` package includes my "solution" to the PrototypeD assignment, including an example **about** page. You are welcome to use my designs; in fact, I recommend it this time due to the increased complexity of the codebase. If you decide to use your own, carefully replace the widget, layout, and menu code with the code from your solution. Be especially careful to preserve the new code in `CollectionPane` that loads FXML, replaces **summary** widgets, and adds a `CoverFlow` to the layout above the **table**. Most other files are unchanged from before.

For the FXML+FXCSS part of the assignment, your goal is to reproduce the layout and styling of the **summary** from PrototypeB (mine or yours, as you prefer). The places to add code are:

> #01a (in `SummaryFXML.java`) — Add and name members for each of the widgets loaded from FXML for use in the **summary** portion of the Collection Pane.
>
> #01b (in `CollectionPane.java`) — Get the FXML references to the **summary** widgets and assign them to corresponding widget members in `buildMovieViewUsingFXML()`. (Also read the note about swapping comments in the method just below where you do that.)

#02 (in `Summary.fxml`) — Define the layout of the **summary** using FXML.

#03 (in `Summary.css`) — Define the styling of the **summary** using FXCSS.

For the **coverflow** part of assignment, your goal is to reproduce the layout and styling of your design from DesignE, including the buttons and the general feel of the movie browsing task. The places to add code are as follows:

#04ab (in `CoverFlow.java`) — Add members for your left and right navigation buttons. Create your buttons and add them to the base pane on top of the other layers.

#05ab (in `CoverFlow.java`) — Register and unregister each of your buttons with the `ActionHandler`.

#06 (in `CoverFlow.java`) — Make any necessary updates to layout and styling of buttons.

#07ab (in `CoverFlow.java`) — Implement button handling in the `ActionHandler`, then implement the equivalent handling of key presses `handleKeyPressed()`.

#08 (in `CoverItem.java`) — Add a member for each element used in your item layout.

#09abc (in `CoverItem.java`) — Implement `createLayout()`, `updateLayout()` and `updateStyles()` to define the default layout of items and update their layout and styling whenever movie properties change.

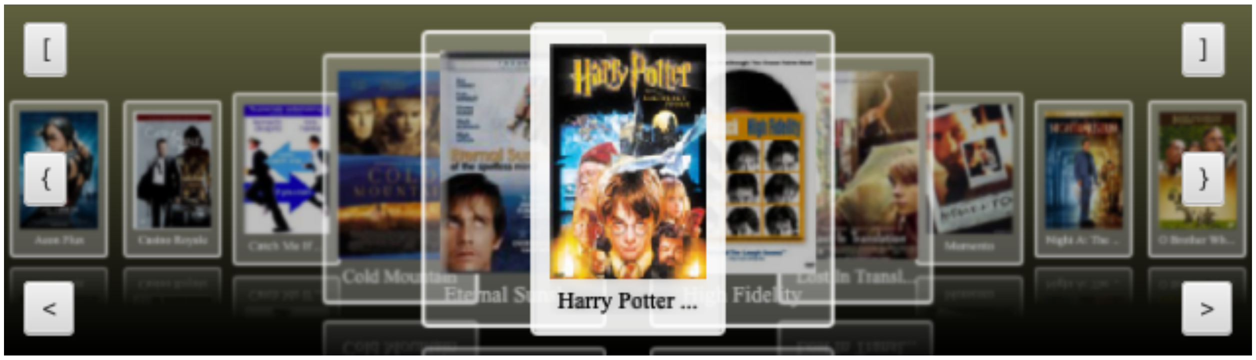
#10ab (in `CoverItem.java`) — For each movie property that you utilize in your item layout somehow, uncomment the corresponding line in `registerPropertyListeners()` and `unregisterPropertyListeners()`.

#11 (in `CoverFlow.java`) — Implement the layout and styling of the flow itself. The provided starter code provides examples of positioning, sizing, and styling items to create a basic flow look and feel.

#12 (in `CoverFlow.java`) — If necessary, rewrite `updatePane()` to put movie items into the flow in the same front-to-back order as in your design. The provided code is likely to work for many if not most designs that put the central item frontmost.

#13 (in `CoverFlow.java`) — Experiment with different keyframe sequences, durations, and easing approaches to get the exact animation look you want for your design. The provided code is likely to be suitable in many cases. Still, animation is a lot of fun to experiment with...

I recommend following the order above. You will probably spend most of your time working on #9 and #11. Keep readability in mind and document your code helpfully. **TEST WITH GRADLE!**



This is my coverflow design, showing only titles and posters. Implement your own design with all six movie attributes!

Turning It In

Turn in a complete, cleaned, renamed, zipped **COPY** of your `PrototypeE` directory:

- Put a copy of your `DesignE.bmpr` file in the `Results` directory.
- Take screenshot of a window showing your *Collection* tab in an informative graphical state.
- Put the screenshot in the `Results` directory as `collection.png` or `collection.jpg`.
- Go into the `ou-cs-hci` directory.
  - Make sure it contains all of the modifications and additions that you wish to submit.
  - Run `gradlew clean` to reduce the size of your build.
  - If you're using an IDE, remove any IDE-specific files (such as the Eclipse `bin` directory).
- Append your 4x4 to the `PrototypeE` directory; mine would be `PrototypeE-weav8417`.
- Zip your entire renamed `PrototypeE` directory (including `About`, `Build`, and `Results`).
- Submit your zip file to the Homework - Prototype E assignment in Canvas.

To score the assignment, we'll be looking at how many elements in the summary and coverflow designs appear as components in your prototype; how well the prototype reflects those designs' layout and style; how clearly your code (JavaFX, FXML, FXCSS) is organized and documented; and the overall quality of experience using your coverflow. The maximum score is 20 out of 20.