# Unit 7 Assignment

## GENERAL

1. Save each class in a separate Java source code file and name it *StuPrefixClassName.java*. Follow the coding style as explained in Unit 1 Assignment.

2. Overall comment your program appropriately (file prolog comment, method prolog comment, and comments on the code). Pay attention to the standard stuff like coding style, meaningful identifier names, indention, and locations of braces.

3. When you’re done with an exercise, include a screenshot of the execution of the program in your assignment report document. This is to remind you to verify the execution result.

## EXERCISES

We will build a GUI course registration program simulating course registration in a university.

Program printouts and screenshots below show data from an example data set. You may get a different data set.

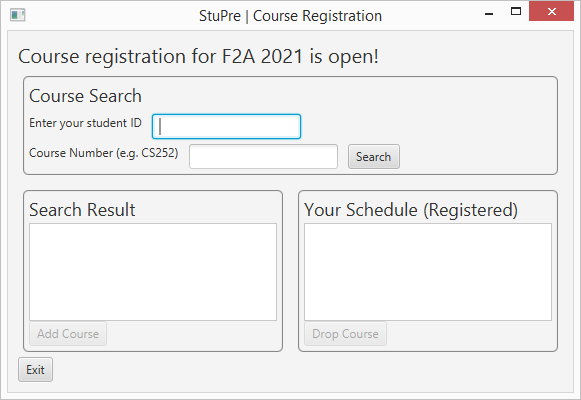


Figure 1. Startup Screen.

* The program is intended for course registration for a specific term.
* There are only CS courses in the course data file. ☺
* At startup, the program connects to course session data and course registration data.
* Users enter a student id (sorry, we don’t have a login screen yet) and a course number. The search button does two things:
  + Search for and load student schedule data (courses this student already registered with) into the schedule area (the area labeled “Your Schedule”),
  + Search for and load course session data matching the course number into the search result area.
* (Optional) The Add Course button and Drop course button are already set up, but don’t have actual event handler code. You’re welcome to play with those.

Build your program following those exercises:

1. Create a project and add the JavaFX user library into the project. Review the tutorial from the last unit if needed.

Add a package named after your last name. For example, student John Doe will name his package doe.

1. We will set up the non-GUI portion code first.

Find the following classes in four HW7xxx.txt files for this assignment:

HW7CourseSession: complete. Do not change it.

HW7RegistrationRecord: complete. Do not change it.

HW7RegistrationManager: complete. You will add testing code to its main(), but otherwise do not change it.

HW7DataSource: partial. Will need to complete one method.

Put them into your project. Each class should be in its own source code file. No need to add the StuPre prefix to any of those four classes.

Add the data folder (two csv files in it: courseData.csv and registrationData.csv) to your machine in a proper place under your project folder. See comments on the two static final strings in HW7RegistrationManager. A csv file can be viewed in any text editor. You should try that first if you don’t know what csv means. If you want to see its content in a nicer format, open a csv file in Microsoft Excel on a Windows computer or TextEdit or TextWrangler on a Mac machine.

Once set up properly, the program should compile, run, and generate those prints on the console:

# of course session records: 30

# of registration records: 0

Take a screenshot of the execution of your program at this point and include it in the assignment report document.

Notify your instructor if you find errors in the “don’t edit” portion. No program is bug free. Please discuss the errors with your instructor before attempting to fix them yourself. This way your instructor will be able to inform the class if needed. Thanks in advance for your help!

1. Study the given code and data to understand how the program is designed. Professional software developers do this daily when they maintain or extend programs built by others.

Answer the following questions about the data program:

* 1. (data files) How is data in the COURSE\_SESSION\_ID column of registrationData.csv related to the columns (and data) in courseData.csv?
  2. (HW7CourseSession class) When does the addStudent() method return false? When does the removeStudent() method return false?
  3. (HW7DataSource class) The loadCourseSessionData() method calls trim() of String on all read data and toUpperCase() of String on specific data. What do the two methods do and what is the benefit of such calls?

* 1. (HW7RegistrationRecord class) How do the two constructors differ from each other?
  2. (HW7RegistrationManager class) List the main tasks accomplished by the loadData() method. How is the compareTo() of HW7CourseSession class used here?

1. (HW7DataSource class) Complete the loadRegistrationData() method following comments in the class. Once completed, should generate the following prints (the displayed invalid course session id may be different):

Error loading registration records: invalid course session id CS335DL

# of course session records: 30

# of registration records: 7

Take a screenshot of the execution of your program at this point and include it in the assignment report document.

1. (HW7RegistrationManager class) Add testing code to the main() following comments in the class.

Take a screenshot of the execution of your program at this point and include it in the assignment report document.

1. (Pick a proper name for your GUI class and add the StuPre prefix). Now we are going to add a GUI front to the program.

Comment out the main() in the HW7RegistrationManager class. Do not delete it, as your exercise 5 work is there.

Add a JavaFX GUI class. Configure the run configuration. Review the tutorial from the last unit if needed.

Set up the GUI first. Follow instructions to copy given code from a GUIPortionCode.txt file for this assignment. Include your StuPre prefix in the window title.

When set up properly, your program will look like the window in Figure 1 at startup.

Find an existing student id from the registration data file (i.e. a student with registered courses) and use it to test the Search button:

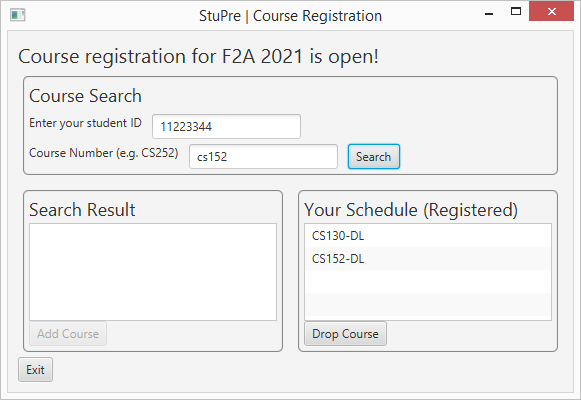


Figure 2-1. Your program should load schedule of this student, but the search by course number result should be empty, as the code is yet to be added.

Take a screenshot of your program window (should show your StuPre) showing searching result like in Figure 2-1 and include it in the assignment report document.

1. Find such a method in your GUI class and complete the method following comments there:

// event handler for search course button

private void searchCourse() {}

Once completed, the program should work like this:

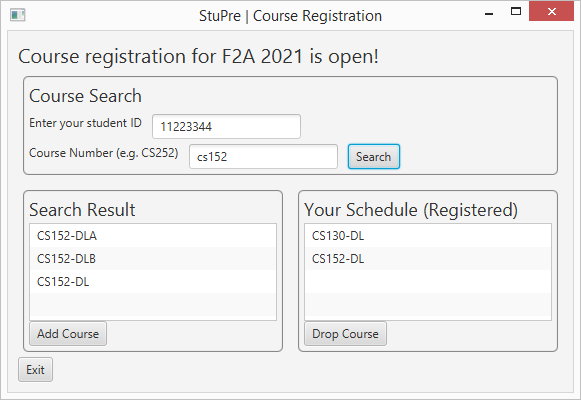


Figure 2-2. Search result after the searchCourse() method is complete.

Take a screenshot of your program window (should show your StuPre) showing searching result like in Figure 2-2 and include it in the assignment report document.

Optional: if you are interested in completing the add and drop functionality, you need to know how to use ListView.

<https://openjfx.io/javadoc/11/javafx.controls/javafx/scene/control/ListView.html>

<http://math.hws.edu/javanotes/c13/s3.html> : the EditListDemo.java program in chapter 13.3.3.

1. Reflection: answer those questions AFTER you’ve completed this assignment:
2. What is the hardest part of this assignment for you? Please explain.
3. When may you use exception handling in your projects? Briefly explain.

## SUBMISSION

Submit five .java files + one word/PDF document. Please put screenshots and answers to questions into your word/PDF document.

* Exercise 2: one screenshot showing the execution result of the given non-GUI portion.
* Exercise 3: answers
* Exercise 4: one required screenshot.
* Exercise 5: one required screenshot. Non-GUI portion done (4 java files)
* Exercise 6: one required screenshot (like Figure 2-1) showing work of completed non-GUI plus given GUI.
* Exercise 7: one required screenshot (like Figure 2-2) showing completed GUI project (1 more java file).
* Exercise 8: Assignment reflection
* Check the completeness of your work against the rubric before turning it in.

## Rubric: Unit 7 Assignment

| **Criteria** | **Ratings** | | | **Pts** |
| --- | --- | --- | --- | --- |
| **Exercise 1**  **(package setup)** |  | 0.5 pts. Correct package statement in source code. | 0 pts. Incorrect or no submission. | 0.5 |
| **Exercise 2**  **(execution of given**  **non-GUI portion)** |  | 0.5 pts. Correct execution result (screenshot). | 0 pts. Incorrect or no submission. | 0.5 |
| **Exercise 3**  **(answer questions)** | 3 pts. Proper answer to a-e. | 2.5 – 0.5 pts. Incorrect or missing response to at least one question: 0.5 pts for each of a-d, 1 pts for e. | 0 pts. Incorrect or no submission. | 3 |
| **Exercise 4**  **(HW7DataSource)** | 5 pts. Correct method meeting all requirements. | 4.5 – 0.5 pts. At least one requirement is incorrect or missing:  (1 pts) try-with-resources statement to open input file (and close resources)  (0.5 pts) loop to read all records, each record as a string;  (1 pts) extract data from a record and check for empty strings;  (1 pts) create a proper object from the data and add the created object to the local arraylist;  (0.5 pts) catch IllegalArgumentException with proper prints;  (0.5 pts) catch DateTimeParseException with proper prints;  (0.5 pts) catch IOException with proper prints. | 0 pts. Incorrect or no submission. | 5 |
| **Exercise 5**  **(HW7RegistrationManager)** | 2 pts. Proper code added to main(). | 1 pts. At least one requirement is incorrect or missing:  (1 pts) call a proper method to add a registration with the same data twice and get expected results;  (1 pts) call a proper method to remove registration with the same data twice and get expected results. | 0 pts. Incorrect or no submission. | 2 |
| **Exercise 6**  **(given GUI)** |  | 1 pts. Correct GUI class setup so the program works. | 0 pts. Incorrect or no submission. | 1 |
| **Exercise 7**  **(Event handling)** | 5 pts. Correct event handling method searchCourse() meeting all requirements. | 4.5 ~ 0.5 pts. At least one requirement is incorrect or missing:  (0.5 pts) check for empty course number,  (0.5 pts) report with an alert,  (0.5 pts) focus set to course number input GUI control, and  (0.5 pts) end the method;  (1 pts) call a proper method to search by course number, and  (1 pts) save returned result in a proper instance data member;  (1 pts) call a proper method to initialize the search result list view. | 0 pts. Incorrect or missed all requirements; no submission | 5 |
| **Exercise 8. Reflection** | 1 pts. Answered both questions. | 0.5 pts. Answered only one question. | 0 pts. Didn’t answer the questions or no submission. | 1 |
| **Required program name (StuPre part) + Style Points (Proper comments; meaning identifier names; consistent indentation)** | 2 pts. Correctly named program and proper style in all three areas of style points. | 1 pts. Problems in one of the following areas: program name, three areas of style points. | 0 pts. Problems in two or more of the following areas: program name, three areas of style points. | 2 |
|  |  |  | *Total Points* | 20 |