Karlsruhe Institute of Technology

aifb
Institute for Applied Computer Science
and Formal Description Methods

Prof. Dr. D. Ratz

Philipp Stegmaier (philipp.stegmaier@kit.edu)
Helen Schneider (helen.schneider@kit.edu)
Ferdinand Mütsch (muetsch@kit.edu)

Programming commercial systems - Applications in Networks with Java

summer semester 2023

submission form

(Processing in the period from June 5, 2023 to June 25, 2023)

1 Task

You often and happily eat in the KIT cafeteria. One Wednesday afternoon, you decided that you would like to get a better overview of your meals. Write aexecutableJava program - the “Mensa Food Tracker” -that supports you in this. It should allow you to record the dishes you have chosen from the current weekly menu in the form of a meal history and to display various key figures, in particular costs and nutritional values, in an aggregated manner. The program should have agraphical user interface (GUI)so that even people with little technical knowledge can use the program.

The program is open to all studentsindividuallyand must meet the following requirements:

1.1 Functional requirements

- There should be aList of all offered canteen dishesfor a user-definedday to be selecteddisplayed if data is available for that day. Thedefault valueWhen starting the program you should always click on thecurrent dayThe list should contain at leastnameandPriceof the respective dish are displayed, optionally and if desired also other attributes. You will get aexternal library provided with which you can request the required data. More on this in section 1.5.
- Users should select a dish from the list and add it to theirfood historyLikewise, courts should be drawn from historyremovedcan be.
- TheEssen historyis a list of all dishes recorded so far, (descending)sorted by date.For each dishName, Date andPrice,and at least three of the presentnutritional informationbe displayed. There can be several dishes in the history per day (even the same dish several times). If no dish was recorded for a day, this day should not be displayed in the history.
- Furthermore, there should be an overview of the variouskey figuresof all dishes in the history. This includes: totalEnergy intake in kilocalories (kcal),the total amounts ofprotein, carbohydratesandFat,each in grams (g), theTotal costs in Euro (€),and the Veggie share in percent (%),ie the ratio of vegetarian or vegan dishes to all dishes in the history. If a dish is added to the history or another is removed, the key figures should automatically updated,ie recalculated.

- So that theie H istori ealso about is maintained across multiple program starts, the state of the application should be automatically eimCloseof the program into afile savedand the next onestartof the program again from thisfile readIf no file is available when initially used, the program should start with an empty history. Details on the file format to be used can be found in section 1.4.

- In summary, the application inthree main areasbe divided into:
  - The first area contains components toSelect day (date),the correspondingView menuand individual dishes of historyaddto be able to.
  - The second area includes theHistory,ie essentially a list of all dishes already added sorted by date, and the option to delete them againremoveto be able to.
  - The third area shows the aforementioned aggregatedkey figuresgraphically appealing.

1.2 Non-functional requirements

- UseJava 11or newer.
- When implementing your application, please follow the steps presented in the lecture.MVC approach.
- For better understanding, all publicly visible classes and methods of your program should be labeled with useful JavaDoc commentsbe provided.
- Theexpected error cases,that a file cannot be read or written or that no data can be retrieved for a requested day,treatedand the user addresses the problem (e.g. usingJOptionPane)pointed outbecome.
- The user interface should remain appropriate even with varying window sizesoperableIn particular, all list elements should always be visible (e.g. by scrolling).

1.3 Notes

- The task is deliberately formulated in an open manner. You can let your creativity run wild when solving the task - especially when designing the interface. Everything that is not explicitly specified is up to your interpretation. It is important that the requirements mentioned are met. But you are welcome to go beyond them!
- Please do not use any external libraries during implementation other than those explicitly provided to you.
- Pay attention to aeasily readable source code,In particular, structure your code clearly and use meaningful identifiers for the variables.
- For the graphical interface, you can use all components from Swing and AWT. GUI editors such as *WindowBuilder*are not excluded in principle. However, if you have problems using them, you are on your own.
- To save the current state as aCSVto save (see section 1.4), for example WindowListenerbe used.
- For example, the date selection can beJSpinner.DateEditor.
- Danger:A program that does not compile has almost no chance of passing the test. Make sure that you submit a functional program. Commented out code is also generally not evaluated.
- Danger:We will use software to detect possible plagiarism. This software analyses the control flow of your compiled source code and cannot be tricked by renaming or moving classes, variables, methods or similar. If plagiarism is present, we will not determine who copied from whom. Both submissions will then be considered as at least*failed*rated.⇒Do not copy from other students or distribute your own solutions.

1.4 File format

To keep your food history even after closing the program, it must be saved persistently in a file. UseComma-Separated Values (CSV)as a file format. This is a very simple, tabular format in which individual entries (here: canteen dishes) are stored aslinesand their attributes (e.g. price)ascolumnsColumns are separated from each other by an arbitrary, but previously defined symbol - usually a comma (,). Use aSemicolon (;) as a separator.In addition,
CSV files usually have aheaderin the first line, which gives the names of the columns for better readability. A simplified example could look like this:

```
name;price;kcal;type
Pasta in tomato sauce with bacon, peperoncini and grated cheese;3.2;943.0;PORK
Pasta in tomato - broccoli sauce with grated cheese;3.2;856.0;VEGETARIAN
```

Implement methods that allow you to create objects of the classMensaMealto write into a CSV file (“serialize”) and to read from it again (“deserialize”).

Tip:You can use the instance methodsplit()the classstringuse to create astringusing a separator into severalstring components.

1.5 External library:mensascraper-lib.jar

To query the cafeteria menu, we provide you with an external library asJAR fileIt can be downloaded as a .jar file and integrated into Eclipse (or other IDEs). You learned how to do this in the computer lab P00.

The library only provides a public method that can be called up with a cafeteria to be queried (here: Cafeteria at Adenauerring) and a date and provides a list of all the dishes offered on that day (MensaMeal)Use the JavaDoc documentation and the example provided to familiarize yourself with the use of the library and the classes available.

- Download:https://github.com/muety/kit-mensa-scraper/releases/latest
- Documentation:https://docs.muetsch.io/kit-mensa-scraper
- Example:https://github.com/muety/kit-mensa-scraper#usage-example

1.5.1 Notes

- The Student Union does not provide historical data, ie it can only provide the menu of the current day or higherIn particular, for example,*not*yesterday's menu is available. If no data is available for a day, an empty list is returned. If a day in the past is requested or if retrieving the data fails for other reasons, auntested MensaScraperExceptionthrown.
- Using the library requires an activeinternet connection.If this poses a serious obstacle for you, please contact the instructor.
- The library uses the Java 8-basedjava.timeAPI, especiallyLocalDate.In Swing / AWT, however, the older java.util.Dateclass. A quick internet search should explain the conversion between classes.

2 Delivery of the service

- The finished program is submitted asArchive file, i.e. as a zip file,via ILIAS. Instructions on how to export projects in Eclipse can be found on the computer lab sheet P00.
- The submission takes place under the exercise object*mandatory tax*via the buttonSubmit file.Rename the file with “Lastname_Firstname_and-Abbreviation_Submission.zip” (e.g. “Mustermann_Max_uabcd_Abgabe.zip”). Taxes that cannot be assigned to a person arefailedrated.
- The deadline for submission isJune 25, 2023 at 11:55 p.m.Submissions not made on time will befailedIf you experience any problems uploading, please contact your tutor or the instructor directly. The deadline stated here also applies.

3 Open questions?

- For content and organizational questions regarding the submission, please contact the*Forum for questions about delivery* to use.
- For basic questions about Java, please feel free to contact the*forum for content-related questions*and contact your tutors.

Each student must solve the task individually. Attempts to cheat will result in all students involved being assessedfailedand are reported to the examination board.