4CCS1PPA Programming Practice and Applications

Coursework 4: London Property Marketplace

Michael Kölling and Jeffery Raphael
Questions? programming@kcl.ac.uk

Your task is to develop a GUI for exploring properties that are available for temporary rental in London. You will be working on a real world data set of the London property market, from AirBnB. This data is provided to you through existing Java classes that are available from the KEATS page. You should take some time to familiarise yourself with the methods that you have access to through these classes.

---

1 GitHub Repository

One member of each group is responsible for the following.

- Selecting a group name (humorous names are welcome, but groups with rude names will be penalised!).
- Creating a central repository to be shared by the group on the KCL GitHub Enterprise system here: `https://github.kcl.ac.uk/`.
- Creating a text file in the repository with the group name as the title, and the group name as the first line of the file.

Initialising the Git repository.

Each group member must then do the following.

- Create a development branch on the repository.
- Edit the branch by appending their name to the text file.
- Merge the branch with the master branch.

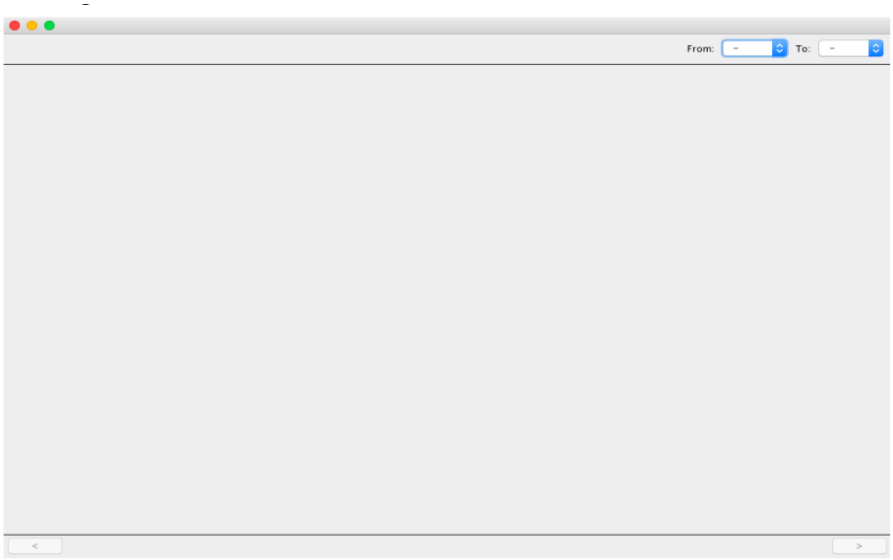
2 Register Group

Each group member must complete the Assignment 4: Group Selection quiz. This will give you access to the submission links. If at anytime your group members change, please update your responses. The quiz will be available for the duration of the assignment.

3 Base Tasks

Application window

The application is designed to process and display the data from the database in a digestible form for a user. To do this, we will create a multi-panel and multi-window application. The first, and main, window of this application should look similar to the following example.



The window is designed to hold a series of different panels, each of which contain data relating to the retrieval and display of data from the API. Details of what these panels should contain is given in the following sections of this brief. Key behaviour that should be offered by this display is as follows.

- There should be the ability to move left and right through the panels contained in the centre of the display using "back" and "forward" buttons in appropriate positions. In the example, these are shown in the bottom left and right. If there are no additional panels forwards/backwards then the button should cause the display to loop round to the first/last panel.
- The top right of the frame should feature two drop-down boxes, appropriately labelled, allowing a user to select a price range for the properties they want to see statistics about.
- The user should be alerted if they have selected a price range that is invalid, such as the "from" price they have entered is greater than the "to" price.
- The "back" and "forward" buttons should be initially disabled, until the user has selected a price range. This is because the other frames available are going to process and display the data loaded when a user selects a price range, and are thus initially empty. The first panel, however, is unconnected to the data, and should thus be shown to the user when the frame first loads, as discussed in the next section.

Panel 1: Welcome

Having a user land on a blank screen like the one shown in the image above isn't a good design. Instead, the first panel loaded into the main window when a user loads the application should welcome the user, and give them instructions on its basic use.

Once a prince range is selected by the user, the first window should additionally show which price range is currently selected.

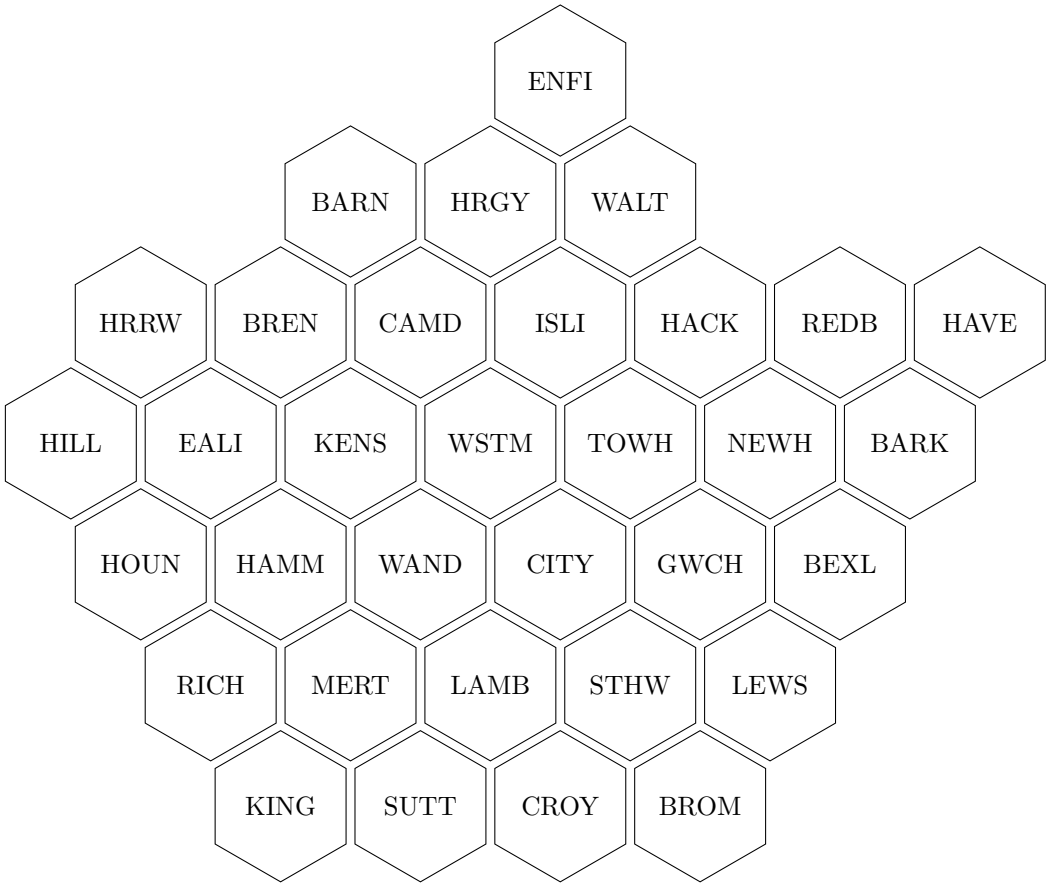
Panel 2: The Map

The first data related panel that should become available for access by the user once they have selected a date range is a panel that visually demonstrates the geographic content of the data set.

This should be done by showing the user a map of London's boroughs on the panel. Each borough on the map should be clearly labelled with its name.

Further, for each borough, there should be some visual indication on the map of how many properties are available in that borough. For example, the visual indication could be done by adding markers to each borough where a large marker indicates there are more properties in the borough. Or you could colour each borough depending on the number of available properties, where the greener the colour the more properties are available.

Feel free to make use of the graphic below if you want to (a PNG version is included in the project folder from KEATS). You may use some other visual representation or map if you want to, but your visualisation should be somewhat geographically accurate.



List of properties

Upon a user selecting a price range, and after the appropriate visualisation of property availability is displayed, it should be possible for a user to click any of the boroughs in order to see more about the properties in that specific borough. This information should be presented in a new window, which opens when a borough is clicked. The title of the window should include the full name of the borough that the user has selected.

The window should also display a simple list of available properties with the following details.

- Name of the host of the property.
- Price of the property.
- Number of reviews per property.
- Minimum number of nights that someone can stay at the property.

The window should have an additional drop-down menu that allows the user to select whether to sort the list by number of reviews, price, or alphabetically by host name. Making a selection from the drop-down menu should reorder the list automatically.

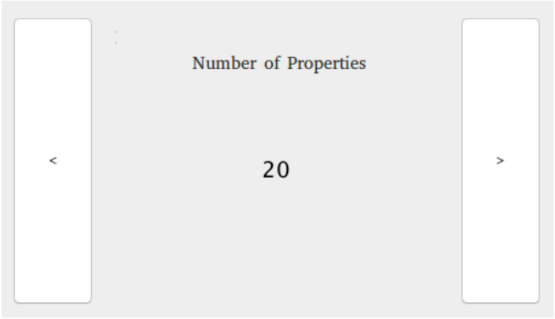
When one of the properties is clicked, the description of the property should appear in a new window.

Panel 3: Statistics

This panel will present a series of statistics, based on the information derived from the data set, and should thus be available at the same time as the map to be observed by the user, should they navigate to this panel. We will derive eight statistics over the data available.

Panel behaviour

The panel should be separated into four distinct section, which we will refer to as statistic boxes. Therefore, at any given time, only four of the available eight statistics are shown. Each statistic box should look similar to the following, with each box displaying a different statistic.



Using the buttons shown, the user should be able to click between the different available statistics, in a similar manner to the way to which they are able to click between different panels of the main window.

At no point should the same statistic appear twice on the panel. Instead, when the panel is first shown, four different statistics should be selected. Moreover, when the user is clicking between the different statistics, it should not be possible for them to configure the panel such that it shows the same statistic in more then one box.

Statistics

The base statistics you need to implement are as follows.

- Average number of reviews per property.
- Total number of available properties.
- The number of entire home and apartments (as opposed to private rooms).
- The most expensive borough.
  - The most expensive borough should take into account the minimum number of nights. For example, a property A with price £25 with 3 minimum nights (£75) is more expensive than a property B with price £30 with 2 minimum nights (£60).

You should also implement four additional statistics that a user can scroll to in any of the statistics boxes. The statistics should be significantly different to those above.

To get full marks for your additional statistics you need to implement statistics that are insightful and technically impressive — for example, you could take some user preferences to inform your additional statistics, or perhaps you could even use a second data set.

Unit Testing

You should provide suitable unit tests for **one** of the classes in your project. Your testing should be thorough and appropriate. You should not select one of the classes that were provided by us. You should pick a class of your own that is complex enough to warrant significant testing, e.g. one that provide some core functionality of your project.

4 Challenge Tasks

Once you have completed the base tasks, you can implement a fourth panel.

Panel 4: Surprise us!

The final panel is your blank canvas, designed for you to make your program do something creative and interesting, that it doesn't do in any of the other panels. This can include launching new windows, if you would like to. Be as creative as you like, but make sure your functional addition is clear. Only one functional addition is required per group.

5 Submission

The submission consists of three parts: your code repository, a group report documenting the software, and an individual report.

The code

You have to submit a Jar of your project to the "Assignment 4: Code Submission" link in the Assignment 4 section on the PPA KEATS page, before the due date. **The Jar file must contain your source code, i.e., the *.java files, and runs on BlueJ**.

Group Report

Submit the group report to the "Assignment 4: Group Report Submission" link in the Assignment 4 section on the PPA KEATS page, before the due date. The group report should include the following.

- A description of your GUI, including the functionalities provided by the GUI.
- A description of your additional statistics.
- A description of the functionality provided by your fourth panel.
- A description of your unit tests.
- Known bugs or problems. Note: for a bug in your code that you document yourself, you may not lose many marks, maybe none if it is part of the challenge task. For bugs that we find that you did not document you will probably lose marks.

The group report should be no more than four pages long. The group report must clearly state the names and student numbers of all students who worked on the submission.

Individual Report

Each member of the group should have their own individual support. Submit the individual report to the "Assignment 4: Individual Report Submission" link in the Assignment 4 section on the PPA KEATS page, before the due date. The individual report should include the following.

- If you are happy with the contribution of all your group members the report can just be a single line statement.
- If there a group members who you feel were uncooperative or did not contribute a fair amount of effort, you can describe the issues you were having with that group member in the report. You can include any evidence that supports your case. Remember, we will have access to the Git logs.

The same code and group report must be submitted via the assignment 4 submission links on the PPA KEATS page, before the deadline, by **all members of your group**. The individual report should also be submitted before the deadline.

6 Deadline

After submitting your work on KEATS, check that (1) your Jar file and reports (pdf) has been successfully uploaded and (2) that the Jar file contains all of your source files. This assignment (code and report) is due **Wednesday, March 31st, 23:59**.

7 Marking

Applications are marked based on four categories:

1. Program Correctness —The application meets all of the program specifications, i.e., the student has completed all of the base tasks including following submission instructions (e.g., the student submitted a Jar file of their BlueJ project).

2. Code Elegance —The application is written in such a way that the code is reusable and efficient (i.e., memory usage and complexity). The application appropriately uses loops and functions to reduce code complexity and/or repeated code. The application does not have hard-coded solutions or poorly designed solutions. A poorly designed solution is overly complicated, utilises excessive amounts of memory or utilises a slower approach to a problem.

3. Documentation —The application is sufficiently documented. Good documentation/comments should explain what the code does and how it does it. Comments can also be used to highlight nuances in your solution, e.g., a segment of code that only works under certain conditions.

4. Readability —The application is easy to understand and uses good programming practices.

Additional details on how marks are given can be found in the Marking Rubric. Once your submission is marked and written feedback has been returned to you, you will have the opportunity to get additional verbal feedback on your submission from your lab TAs.