PPA Assignment 14

Overview

Do you think there is a problem with any of the content below? Let us know immediately at programming@kcl.ac.uk.

Read through this brief carefully before starting your attempted solution. Also ensure that you comment your code.

You are not advised to print this assignment, but instead to always access the latest version of this brief through KEATS, which will evolve with minor clarifications and corrections throughout the assessment period. Students will be notified of any major modifications to the brief by email.

A partner from your lab session should be selected for this piece of work at the next available opportunity, typically your next lab session. You must not complete any of the assignment below without your chosen partner present, as doing so is likely to jeopardise your grade.

For this piece of coursework, you may work with someone you have worked with previously, if you wish to, but selecting a new partner will be beneficial for your continued exploration of different approaches to programming.

Aims

The aims of this piece of coursework are as follows:

- To use regular expressions to solve practical problems;
- To continue working with components, layout manager and events.

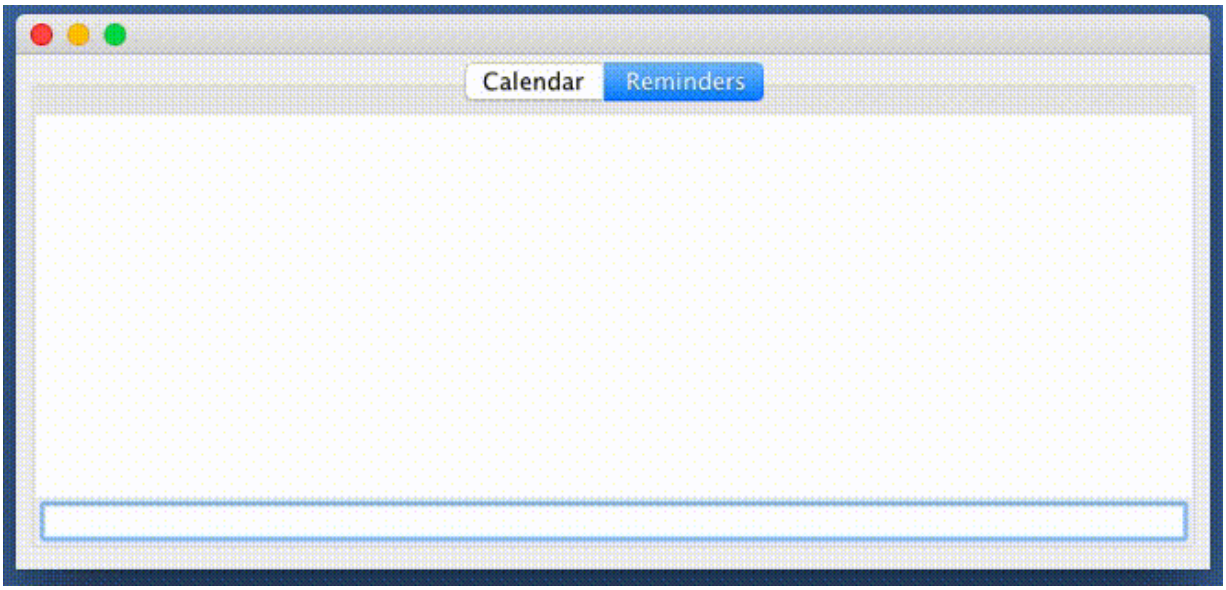
As with many of the tasks this semester, and as part of your development as programmers, you will be expected to conduct some additional research, using the Java API, in order to complete some parts of this assignment. However, none of the tasks in this assignment are entirely unrelated to content taught directly in the course.

Task Overview

Entering information into a calendar, such as the name, time, date and location of an event, or entering information into a list of reminders, can be tedious, especially under the traditional types of UI offered by major companies.

To respond to these limitations, companies like Fantastical are introducing natural language processing into their calendar and reminder applications, in order to allow users to enter details of events, or to add reminders, in natural language. The application then processes this natural language, in order to extract information pertinent to the event, which can be added to other services such as shared calendars.

In this piece of coursework, we will replicate such as an application as follows:

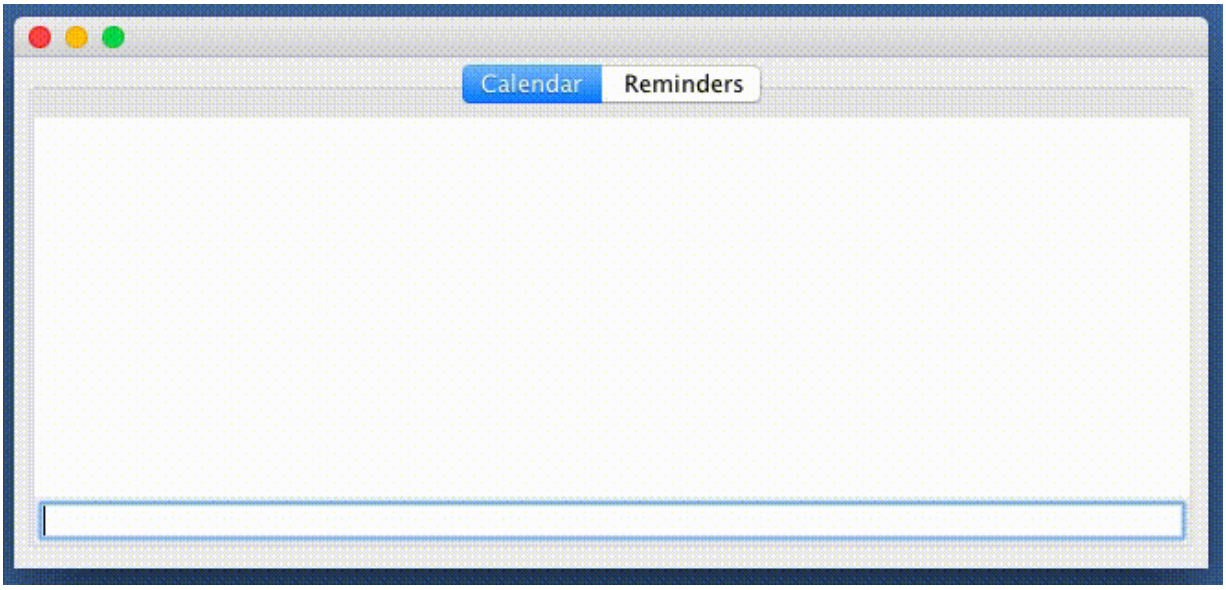


😇 .

Your application is required to do the following:

1. The main window should present two tabs (more information here), labelled calendar and reminders.

2. Each tab should contain a list and a text box. Into the text box, the user should be able to enter natural text describing either an event, or a reminder. This text should then be processed, and displayed in the list:



3. Once the natural text is entered into the text box, the user should be able to press return in order to begin the parsing process, and display the results in the list.

4. The information in the list needs to include: the details of the event (Event), the date of the event (Date), the time of the event (Time), and the location of the event (Location).

5. The format of the date shown in the list (output) *must* be: [Day of Week] [Date] [Month], as shown in the GIF above. The date of the event is identified by a suitable date string (input). Examples of the date formats that *must* be matched by your program are as follows, but you are strongly encouraged to make your program as receptive to different date formats as possible:

    1. Wednesday 8th March
    2. 08/03/2017
    3. On Wednesday (the date of the next day that occurs with this name).
    4. Next Wednesday (the date of the next day that occurs with this name, plus 7 days).

6. The format of the time shown in the list *must* be the time in 24 hour format, with the hours and minutes separated by a colon. The time of the event is identified by a suitable time string. Examples of the time formats that *must* be matched by your program are as follows, but you are strongly encouraged to make your program as receptive to different time formats as possible:

    1. 13:00
    2. 1pm

    3. Evening (enters the time as 20:00)

    4. Morning (enters the time as 09.00)

7. If any information (date, time or location) is not included in the natural text entered, a dash should be shown instead of that information.

8. We will assume that the location of an event is always identified by the prefix `at'. The location itself is the single word following the word `at'.

9. The details of the event are described by any text that is entered that does not refer to the date, time or location of the event.

10. A reminder is identified by the prefix `Remind me to' and should simply be displayed in the list on the reminders tab as this text minus the `Remind me to' prefix, as shown in the first screenshot.

    1. If a reminder contains time or date information, this should also be parsed, as with the event information, and displayed in the list, along with the reminder, in a similar style to the event information. If this information is not included, this can be omitted, as in the first GIF.

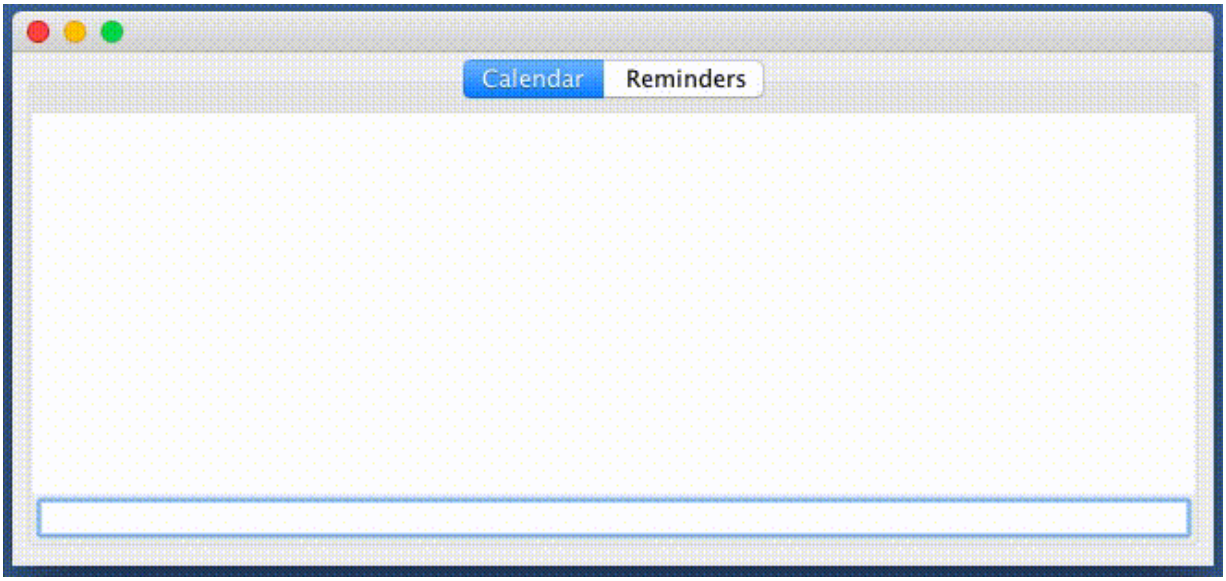
11. In order to remove an item from the list, the user should be able to double click on an entry in this list.

12. Any calendar entries or reminders that are entered into the application must to be visible when the application is reloaded.

Considerations

1. Your application must be implemented in accordance with the model-view-controller (MVC) paradigm.

2. [Update] Remember, your application must have the flexibility to find any of the input information no matter where it appears in the string entered by the user, as illustrated by the following additional example:



3. Which class(es) can you extend in your model in order to store the information held by your program, and add additional features such as parsing and storage?

4. It might be tempting to extract some of the functionality for this assignment directly from online resources like *StackOverflow*. We don't want to strictly forbid the use of resources like this, but remember you should only use these resources to inspire your code, rather than copying code directly, as latter constitutes plagiarism, and may affect your ability to explain your code during your lab assessment.

5. How will you identify whether a date should be suffixed by `st', `nd', `rd' or `th'?

6. How will you identify a double click on your list? Which listener would be appropriate for this?

Optional tasks

1. Implement a login system for your application, requiring both a username and a password, that keeps your calendar entries and reminders secure. You should be able to change any password you select.

2. As noted in the main body of the brief, try and make your application even more flexible in respect of the different formats of information (date, time and location) entered.

3. Export your application as a jar file, so that it can be run from the command line without requiring distinct compilation and run processes. Which command

will you need to run a jar file from the command line?

4. It would be nice if our calendar application could be synchronised with services such as Office 365 and Exchange. You aren't required to implement anything for this task unless you wish to, but consider the services that might be available to allow us to implement such functionality.

Once completed, both you and your partner must submit your assignment using the link marked `Assignment 14: Nexus Submission Link' on KEATS, which will become available after the first pair programming lab session. However, this is not enough to receive a mark for this piece of work. You must also attend the lab session following your submission, so that one of the teaching assistants can mark your work with you present, and ask you detailed questions about it. Revisit the `Lab Assessment and Pair Programming Q&A' guide on KEATS for more information.

Any submitted code that is found to be unduly similar to the code submitted by any other pair(s) of students, will result in a penalty for those involved.

Provisional marks for your code will be released on KEATS within one week of the final lab assessment. Final assignment grades will be submitted to the exam board at the end of the semester.

For all other queries, see the Support section on KEATS, specifically the `Lab Assessment and Pair Programming Q&A' guide.