COSC 2P05 Assignment 3

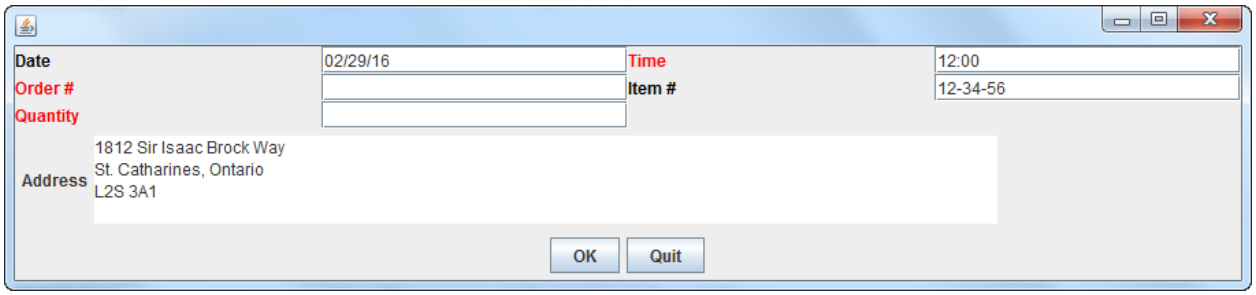
Winter 2015/16

Due: Mar. 21 @ 10:00 am (*Late Date: Mar. 24 @10:00 am*).

In preparation for this assignment, create a folder called `Assign_2` for the assignment. The objective of this assignment is to apply event handling in a problem solution.

Acme Distributing (Version 2)

Acme Distributing has a call center which takes orders from customers. The call center clerk answers a call and takes the order by filling in an order form such as:



When the clerk presses `OK`, the order is placed into a queue to be handled by the warehouse.

Reactive Data Validation

In this improved version of the order entry system, the program should identify data entry errors as soon as they are made (i.e. reactive mode) rather than waiting on the user to press `OK`. Essentially we will be replacing the GUI based on `CheckedForm` with one based on swing using event handling. The rest of the application code remains the same.

Unlike Assignment 2, there will not be a `CheckedForm` class, but rather the main class will handle the form interaction and data validation itself. It should extend `JFrame`, build the form and implement appropriate listener interfaces. It will do the same data validation as Assignment 2. The program should remain fail-soft as in assignment 2.

Each checked field on the form is represented as two parts, a `JLabel` and a `JTextField`. The fields are two across (i.e. label, text, label, text) and take up as much space as necessary. Unchecked fields are represented as a `JLabel` and a `JTextArea` placed below all the checked fields. There are two `JButtons`: `OK` and `Quit` below the unchecked fields.

This can be realized by using a `BorderLayout` for the `JFrame`. A `JPanel` with `GridLayout` can be added to the `NORTH` to contain the checked fields. A `JPanel` with `FlowLayout` can be added to `CENTER` for the unchecked field. Finally, a `JPanel` with centered `FlowLayout` can be added to the `SOUTH` to contain the buttons.

The form should do data validation as the user is typing in the fields. When the user finishes typing in a checked field the form should immediately validate the data in the field and set the color (`setForeground`) of the `JLabel` to `RED` if the field is empty or invalid. This should happen when the field looses focus (`FocusEvent`). A field looses focus if the user hits `tab` or

clicks on any other widget. The checking should also happen if the user hits the `enter` key (`ActionEvent`). Since all checked fields must be filled, the initial state should have the `JLabel`s of all checked fields in `RED`. As the user enters valid data, the color should change to `BLACK` (and back again to `RED` again if it is changed to invalid).

When the user presses the `OK` button, the form should make sure there are no invalid checked fields. If there are invalid checked fields, it will ignore the `OK` button. If all fields are valid, it should log the order, clear the fields and respond to the next order. If `Quit` is pressed, it should handle shutdown and terminate the application.

Much of the code can be borrowed from your solution to Assignment 2.

Submission

Write a simple program to read the order queue file and print the entries to verify that the data is being added. Test your program trying out various combinations/changes to the data entered in the GUI.

Your submission will be in two parts: electronic and paper. The paper submission should include the listings of all Java files you have written. Print the contents of the queue file before and after execution as part of the paper submission.

In addition to your paper submission, you must make an electronic submission of the assignment. You should have a folder for the assignment (`Assign_3`). This folder should include what is needed for the marker to run your program (as indicated below). Zip this folder as `Assign_3.zip`. Log on to Sakai and select the COSC 2P05 site. On the `Assignments` page select `Assignment 3`. Attach your .zip file (e.g. `Assign_3.zip`) to the assignment submission (use the `Add Attachments` button and select `Browse`. Navigate to where you stored your assignment and select the .zip file (e.g. `Assign_3.zip`). The file will be added to your submission. Be sure to read and check the `Honor Pledge` checkbox. Press `Submit` to submit the assignment. You should receive a confirmation email.

The complete paper submission should be placed in an envelope to which a completed coversheet (http://www.cosc.brocku.ca/coverpage) is attached. The submission should be made to the COSC 2P05 submission box outside J328 in accordance with the assignment guidelines posted on the 2P05 Sakai site and not in violation of the regulations on plagiarism (http://www.brocku.ca/webcal/2015/undergrad/areg.html#sec67, http://www.cosc.brocku.ca/about/policies/plagiarism). **Note**: Assignments not including a coversheet will **not** be marked.

DrJava

The `.zip` folder you submit should contain the project folder including all files relevant to the project—the `.drjava`, `.java` and `.class` files for the assignment.

Other Platforms

If you are using an IDE other than DrJava to prepare your assignment, you must include the `.java` source files and the `.pdf` files described above as well as a file (likely `.class` or `.jar`) that will execute on the lab machines. It is your responsibility to ensure that the marker can execute your program.