48024 Applications Programming
Assignment 2: The GUI

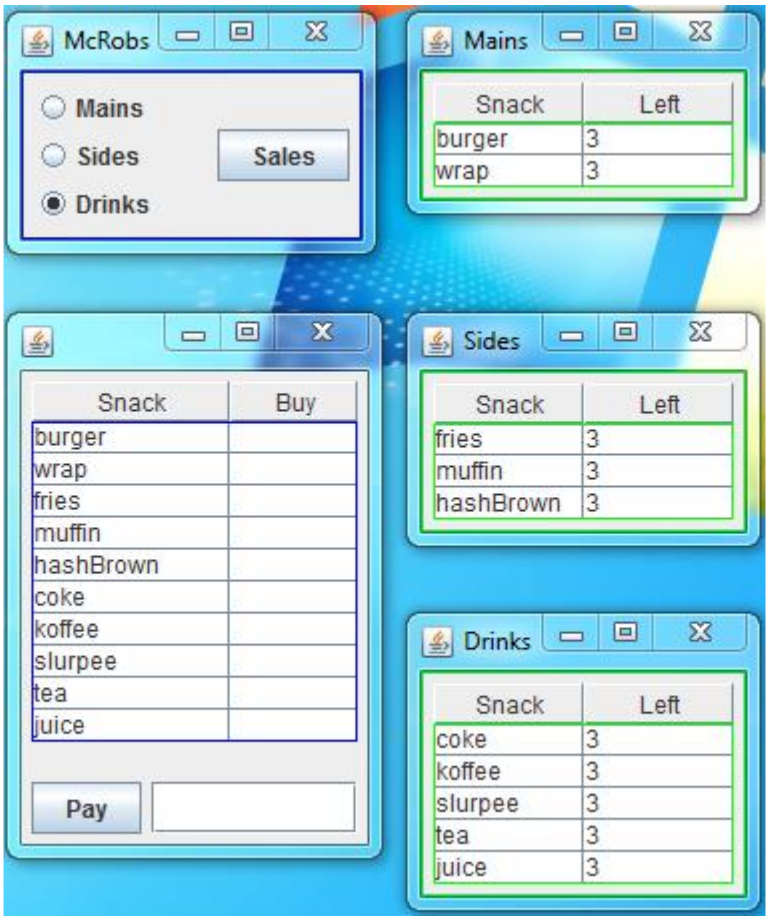
Topics: OO design, GUI, MVC, tables **Value:** 20%
Objectives: This assignment supports objectives 1 - 5.
Due date: Friday 31 October, 6:00 pm.

1. Individual work

You may discuss ideas, approaches, and problems, but you should write every line of code, except for any code provided as base code or lecture notes. For more details, see the Subject Outline or http://www.gsu.uts.edu.au/rules/student/section-16.html.

2. Specification

Build a system that takes an order for snacks. The system has a GUI interface, with the application code stored in a model package. Build your solution so it appears as close as possible to the specified GUI in colour, layout, ansd labelling. The full GUI looks like this:



Mains

Snack	Left
burger	3
wrap	3

Sides

Snack	Left
fries	3
muffin	3
hashBrown	3

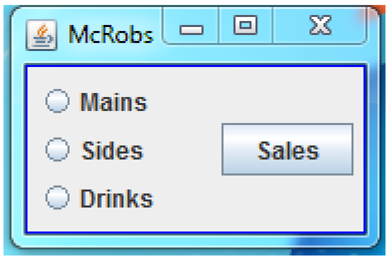
Drinks

Snack	Left
coke	3
koffee	3
slurpee	3
tea	3
juice	3

Snack	Buy
burger	
wrap	
fries	
muffin	
hashBrown	
coke	
koffee	
slurpee	
tea	
juice	

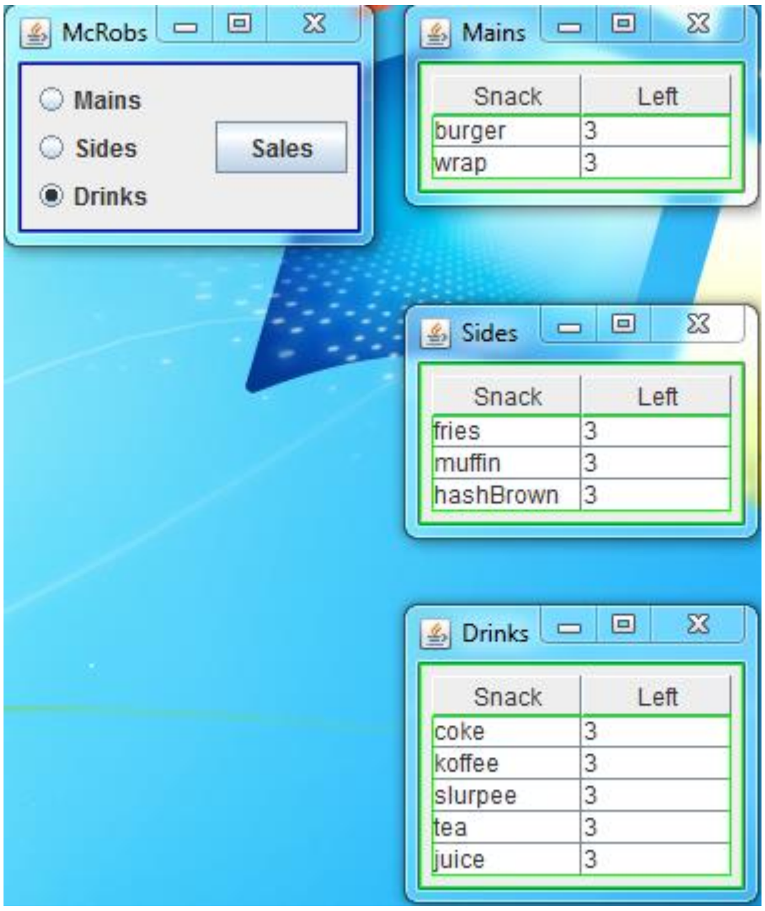
Control GUI

The system starts with the control window (the window at top left in the image above) at location 600, 500. I have called my window "McRobs"; you should choose a short name for your system. Closing the control window closes all the windows in your system.



Show the stock windows

The user selects a snack group, and a table showing the stock on hand for that group appears to the right of the main window, at location 800, 500. Each new radio button selection adds a window 150 pixels below the previous one; a new group window appears every time a radio button is selected. Radio buttons can be selected in any order; the normal order (top to bottom) generates the display shown below; "Drinks" was last selected.



Mains

Snack	Left
burger	3
wrap	3

Sides

Snack	Left
fries	3
muffin	3
hashBrown	3

Drinks

Snack	Left
coke	3
koffee	3
slurpee	3
tea	3
juice	3

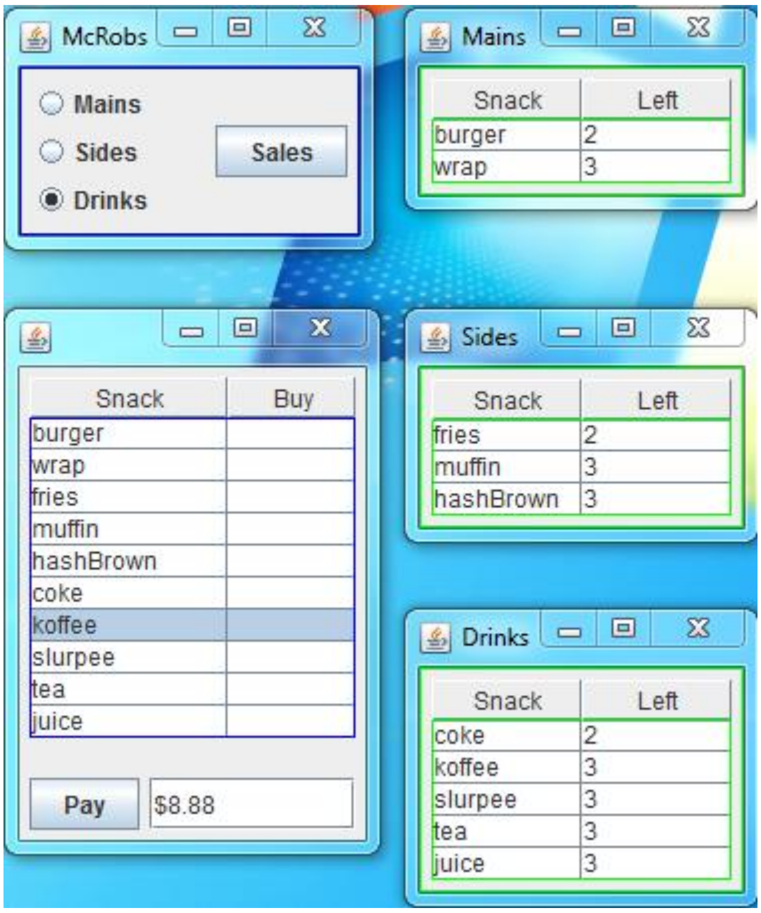
Show the sales window

The user clicks on the `Sales` button, and a window showing a table of all the snacks appears below the control window, at location 600, 650. This ordering GUI is shown below; the second column has a width of 40.

Order window behaviour

The user enters a number of some snack to buy in the right column, and hits `Enter`. The number is cleared, and the cost of the order so far is shown at bottom right of the window. The user repeats as needed to order a meal. Note that the number is cleared and the total updated immediately, when you hit `Enter` in a cell; this is how table entry works. The table below (at bottom left in the image) shows the charge of $8.88 after an order of a burger, fries, and a coke.

Do not check the number to buy. Try buying -43,000 burgers!



Snack	Buy
burger	
wrap	
fries	
muffin	
hashBrown	
coke	
koffee	
slurpee	
tea	
juice	

$8.88

Mains

Snack	Left
burger	2
wrap	3

Sides

Snack	Left
fries	2
muffin	3
hashBrown	3

Drinks

Snack	Left
coke	2
koffee	3
slurpee	3
tea	3
juice	3

The `Pay` button behaviour

When a user has finished their order, they click on `Pay`. This clears the total charge. That's all it does. There is no transfer of money in this system.

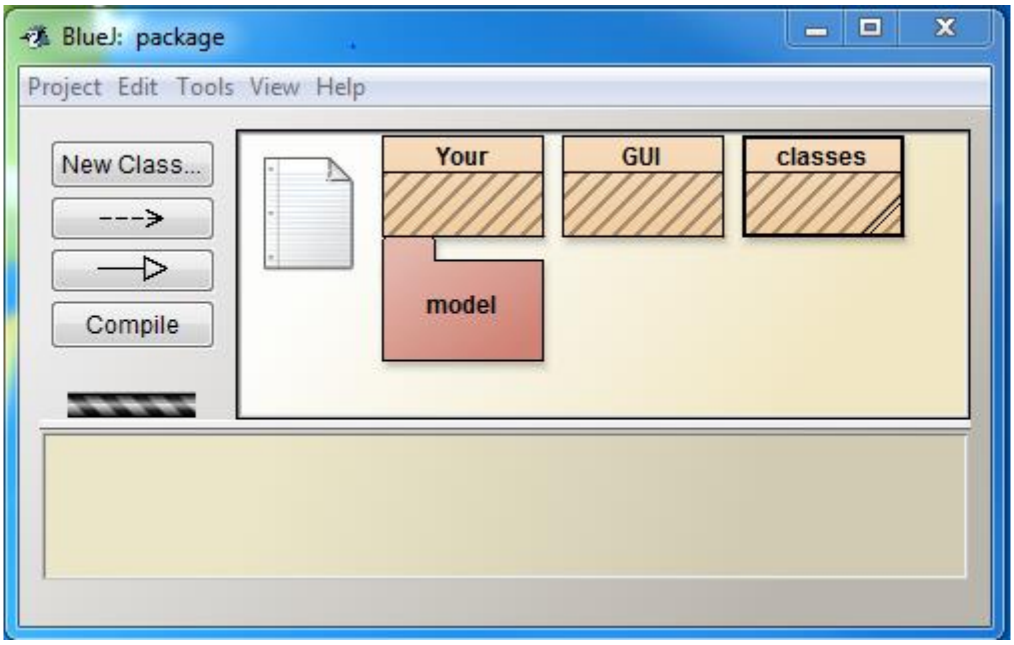
Stock window behaviour

The number of each snack left is shown in a stock window for that group. The number is updated automatically so it changes when the user hits `Enter` for a cell, not when they click on `Pay`. The state of the system is shown at right above, after the order and before the payment.

Package structure

Use the MVC package structure shown below; see lecture 10 and lab 10 for more detail. If you use some other package structure or name, your final mark may be zero. MoMa looks for a model package named `model`; anything else is an error. Place all the GUI code in the top level BlueJ package, as shown below.

The three class names shown in the image are not the names of real classes; do not code three GUI classes named `Your`, `GUI`, and `classes`. The root class must be called `Root`, as stated in Section 3; all other names are your choice.



Base code

The base code is the application code in the package named `model`. You should place classes `View` and `Viewable` in this package, if you use them. You are free to ignore this base code, or to change the code in this package in any way. You are not allowed to change the name of the package; it must be `model`.

3. Submit to PLATE

The root class must be named `Root` and must include a static `main` method for your code to be marked, as shown below.

```
public class Root extends JFrame
{   public static void main(String[] args)
    {   new Root(new Groups());   }
}
```

PLATE cannot mark your solution because it cannot run a GUI; it has no eyes or hands. **You will see no mark** when you submit to PLATE, because PLATE does not run your code. I run your solution manually by downloading the code from PLATE.

PLATE will show you a list of files submitted; if you do not see all your java files listed, then you have not uploaded them. Check that you can run your system through PLATE by clicking on jnlp/jar. If your system does not run when you do this, it will not run when I try to mark it. **Your system must run correctly on PLATE to be marked**.

I cannot submit to PLATE for you. PLATE uses the login id to set the owner of the submission and I cannot log in as you. Please do not ask me to submit your solution to PLATE after the due date, because I can't do it. PLATE does not keep previous versions of your solution, just the latest version submitted. Please do not ask me to mark an earlier version that worked better. I mark whatever you submit as the final version.

Check the format

I run your code on a standard FEIT system. If you develop code on a system that is not the FEIT lab system, such as your laptop, then the GUI may appear differently when run on the FEIT system due to the pixel size. Check that your GUI appears correctly by downloading it from PLATE onto a FEIT system (using `jnlp/jar`) and running it there.

4. Expected work load

This assignment has been estimated at a load of 15 hours for the typical student who has completed all the tutorial and lab exercises.

5. Online support

FAQs (Frequently Asked Questions) and their answers are posted on UTSOnline in **Assignments/2/faq.** If you have a question, check the FAQ first; it may already be answered there. You should read the FAQ at least once before you hand in your solution, but to be safe check it every couple of days. Anything posted on the FAQ is considered to be part of the assignment specification. The FAQ will be frozen (no new entries) two days before the due date.

If anything about the specification is unclear or inconsistent, contact me and I will try to make it clearer by replying to you directly and posting the common questions and answers to the FAQ. This is similar to the job experience, where you ask your client if you are unsure what has to be done, but then you write all the code to do the task. Email ***Robert.Rist@uts.edu.au*** to ask for any clarifications or corrections to the assignment.

I cannot tell you how to design, write, or debug your code; that is your task. I cannot answer questions of the form "Is this right?"; I cannot "pre-mark" your design, or your code. There is no discussion board; the assignment tests your individual ability, so it is not appropriate to share problems and solutions. The tutorials and labs provided a forum for discussion and an opportunity for feedback.

6. Submission and return

There is no scheduled late submission period. An extension of up to one week may be given by the subject co-ordinator before the due date; you have to supply documentary evidence of your claim. **I CANNOT give an extension after the due date.**

You can apply for an extension or for Special Consideration, but not for both. Apply for Special Consideration at *www.sau.uts.edu.au/assessment/consideration.html*. You must provide evidence to support your claim, such as a doctor's certificate, a statutory declaration, or a letter from your employer. SC is designed for students with health, family, or work problems; "I need more time" is not a valid reason for SC, or for an extension.

A file showing the class marks will be placed on UTSOnline about two weeks after the due date; I will send email to notify you. You can also pick up the paper marking scheme from the desk in the FEIT Learning Precinct (FLP): Building 11, level 5, room 300. The FLP is open from 9am to 6pm.

7. Marking scheme

Correctness (except for MVC) is marked on appearance and behaviour. The weight is roughly 30% for appearance, and 70% for behaviour; it varies between tasks.

- appearance: does it look like the snapshots?
  Check the location, size, colour, format, and labels.
  Your solution must appear and work correctly on the standard FEIT system.
  Run it on a standard FEIT system to check that it does appear and work correctly.

- behaviour: does it behave as specified?

10	Initial control window
20	Initial stock windows
30	Initial order window
10	Show and clear running total
20	Update stock windows
10	Correct use of MVC

Minimum essential requirements

A correct solution for more than half of the problem.