

**Assignment Cover Letter****(Individual Work)****Student Information:**

1. Surname
Tantysco

Given Names
Stanley

Student ID Number
2201814670

Course Code : COMP6510

Course Name : Programming Languages

Class : L2CC

Name of Lecturer(s) : 1. Jude Joseph Lamug Martinez

Major : CS

Title of Assignment : Coffee/Tea Shop
(if any) JFrame Application

Type of Assignment : Final Project

Submission Pattern

Due Date : 1-7-2019

Submission Date : 1-7-2019

The assignment should meet the below requirements.

1. Assignment (hard copy) is required to be submitted on clean paper, and (soft copy) as per lecturer's instructions.
2. Soft copy assignment also requires the signed (hardcopy) submission of this form, which automatically validates the softcopy submission.
3. The above information is complete and legible.
4. Compiled pages are firmly stapled.
5. Assignment has been copied (soft copy and hard copy) for each student ahead of the submission.

Plagiarism/Cheating

BiNus International seriously regards all forms of plagiarism, cheating and collusion as academic offenses which may result in severe penalties, including loss/drop of marks, course/class discontinuity and other possible penalties executed by the university. Please refer to the related course syllabus for further information.

Declaration of Originality

By signing this assignment, I understand, accept and consent to BiNus International terms and policy on plagiarism. Herewith I declare that the work contained in this assignment is my own work and has not been submitted for the use of assessment in another course or class, except where this has been notified and accepted in advance.

Signature of Student:

(Name of Student)

1. Stanley Tantysco

Table of Contents

1. Introduction

2. Problems

2.1. Implementations

2.2. How the Program works

2.3. Class Explanation

3. Design

4. Result

5. References

1. Introduction

Recently, there are a lot of shops which sell snacks or drinks as their products. There are 2 interesting points about snack or drink shops. First, staffs of snack or drink shop can manage their customers in queueing. They usually use numbers or customers' name in every order bill. Second, they show the preferences of a specific snack or drink to the customers. They do it because they know every customers has his/her own preferences of their ordered snack or drink. Those are the explanation of my interesting points. So, I decide to create drink shop simulation program which sells coffee and tea as its products.

2. Problems

2.1. Implementations

There were java packages that I used for implementations of my program such as java.util, java.io, and javax.swing. I also used my own built-in package which I named it as DrinkPackage and real life mathematics in my program.

Java.util was main Java sub package in my Coffee/Tea Shop program, which gave containers such as ArrayList as drink order list and Queue as order queue. Furthermore, drink order list could add drink order, delete drink order, get its own list, check its own list whether it was empty or not, and check whether payment was done or not.

Java.io was Java sub package to write drink order list from my Coffee/Tea Shop program into text file which it is named after order queue number. Also, it could read drink orders from text file into the drink order list in program.

Javax.swing was another main Java sub package for designing my Coffee/Tea Shop such as using tabbed pane for multiple panels and designing buttons, labels, text fields, radio buttons, tables, and combo boxes. Furthermore, each buttons, radio button groups, and combo boxes has its own functions to do certain action in the program.

DrinkPackage was my own built-in package that provided 1 class object, 1 abstract class, and 4 interfaces which were implemented and extended to another class objects in my program. They were Drink class object, SizePrefer abstract class, ColdPrefer interface, SugarPrefer interface, Quantity interface, and Topping interface.

Finally, I used real life mathematics to get total price of drink orders in the list and change after payment had been done. Also, I used it to manipulate price of drink based on size, quantity and topping preferences.

2.2. How the Program Works

How the program works can be shown in a flowchart in section three. Now, I explain how detailed my program works.

First, I start the program by running MainMenu JFrame driver class. After it runs, I click Menu Panel. Then, I click one of drink buttons in the Menu Panel. For example,

I click “Cappucino” button in cold coffee section. Its name and price will be shown as “Ice Cappucino” and “40000” in the right down part of panel. Next, I click “Next” Button to proceed into Preference Panel. Because “Ice Cappucino” is cold drink, I proceed to Cold Coffee Panel instead. If I choose hot coffee or tea, I will proceed to Hot Drink Panel instead. Back to the topic, Cold Coffee Panel provides informations such as name, drink type, and price. Also, it shows options such sugar, ice, topping, size, and quantity. For Cold Coffee Panel only, topping options is only group of radio button such as yes or no to Whipped Cream. Meanwhile, Cold Tea Panel only has combobox to topping options. Then, I choose my own preferences for sugar, size, ice, quantity, and topping. If I am not sure of size, topping, and quantity, I click “Check” button to check currently total price. After I choose my own preferences and decide to proceed to Order Panel, I click “Add” button to add my chosen drink into order list and reset form itself. When I proceed into Order Panel, the table show my chosen drinks.

Next, I can click “Back” button to return to Menu Panel, so I can order another drink again. After I am satisfied with my orders, I click “Get Total” Button to get total price of my orders and my order queue number. Then, I fill my payment in payment text field. After I fill it, I click “Confirm” button to complete the payment. After I get the payment notification, I click “Reset” button to clear all orders in table and add my order queue number into queue list.

Last, because this program is accessible to shop staffs only, so I add Check Order Panel in this program. In the Check Panel, there are 1 table, 2 labels and 2 buttons. When I am in Check Panel, I click “Show” button to show order queue number from queue list in label and order list based on order queue number in table. In this way, the shop staffs know what kind of drinks they will make. Whenever they finish making it, they click “Finish” button to clear order list in table and change to next order (pop front the queue number). If I want to close the program, I click “Close Window” button to quit program.

2.3. Class Explanations

There are 2 packages in this final project to do coffee/tea shop simulation. Those are DrinkPackage package and finalproject package as they are mentioned on class diagram in section three. Now, let me explain the details of those 2 packages.

First, DrinkPackage package consists of 1 abstract class, 1 class, and 4 interfaces. Those are SizePrefer abstract class, Drink class, ColdPrefer interface, SugarPrefer interface, Topping interface, and Quantity interface. Drink class has 3 private attributes such as name (String), drinktype (String), and price (double), 2 constructors (default and overloading), 6 methods such setters and getters to private attributes, and 2 overriding setter methods that inherit from SizePrefer abstract class such as setSize(n: String): void and setSizeUp(p: double): void. SizePrefer abstract class has 2 protected attributes such as size(String) and sizeup (double), 2 getter methods such as getSize(): String and ge tSizeUp(): double, and 2 abstract setter methods such as setSize(n: String): void and setSizeUp(p: double): void. ColdPrefer interface has 2 methods such as setice(i: String): void and getice(): String. SugarPrefer interface has 2 methods such as setsugar(i: double): void and getsugar(): double. Topping interface

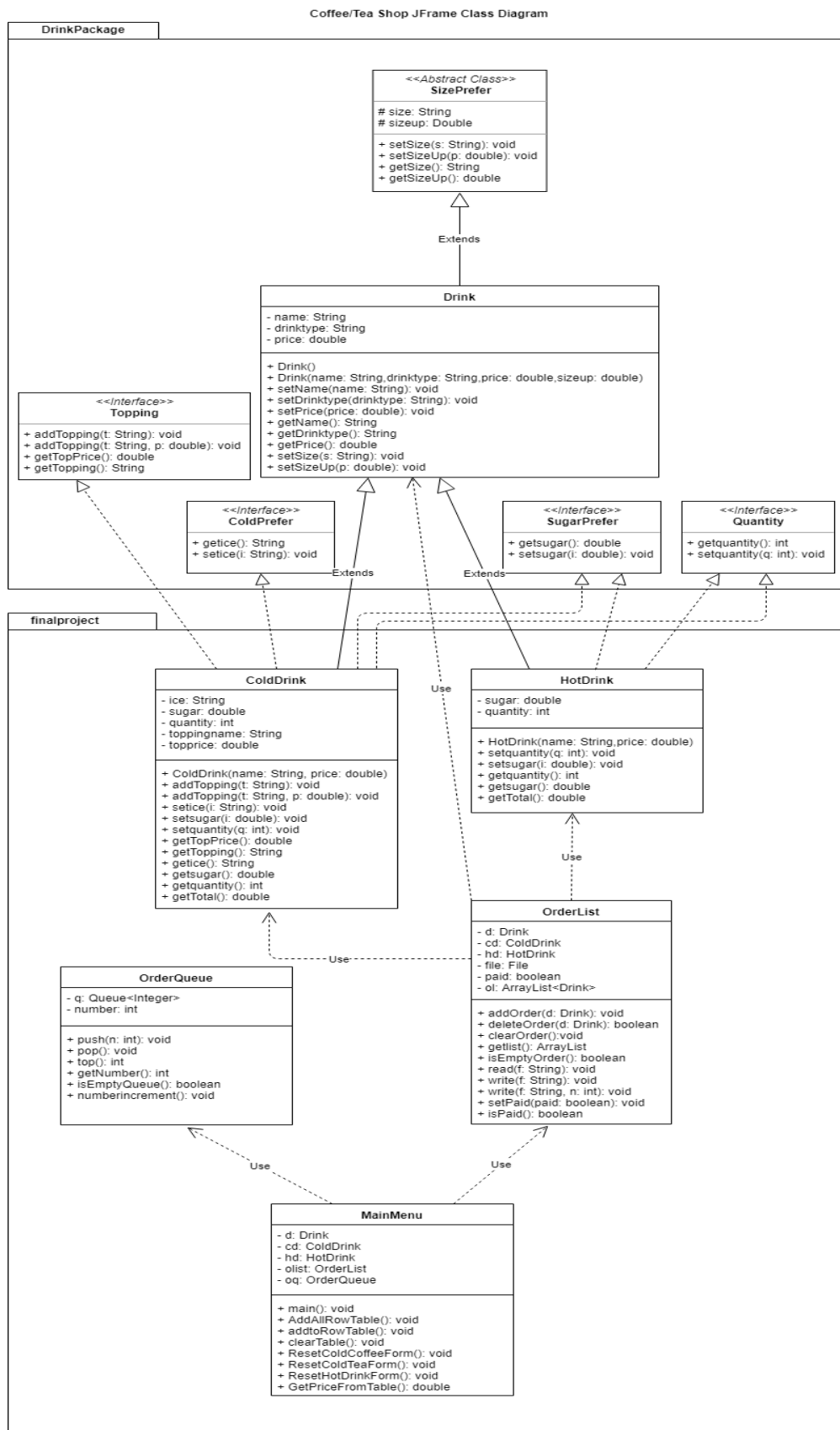
has 4 methods such as `addtopping(t: String): void`, `addtopping(t: String, p: double): void`, `getTopping(): String`, and `getTopPrice(): double`. Quantity interface has 2 methods such as `setquantity(q: int): void` and `getquantity(): int`.

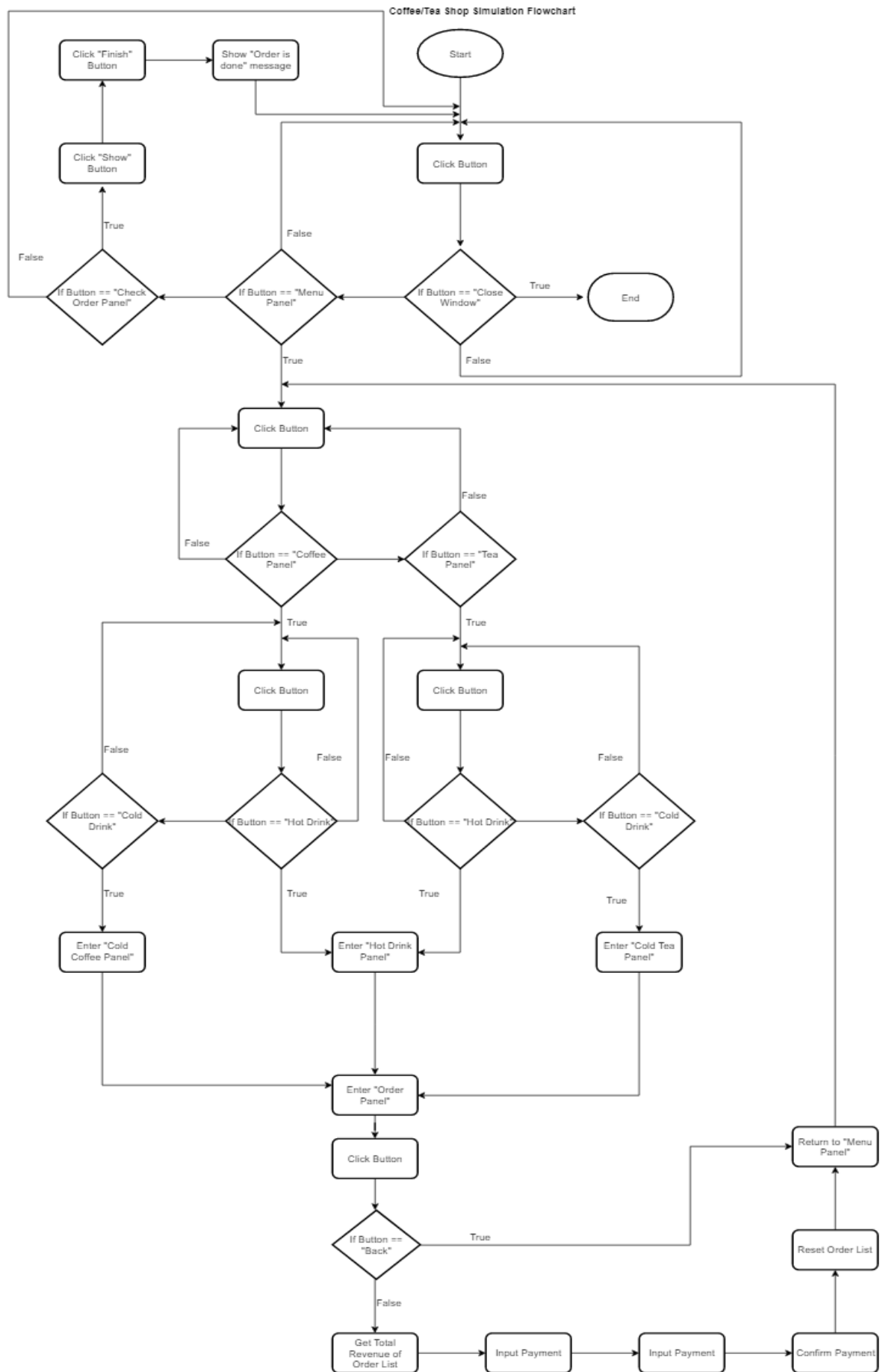
Second, `finalproject` package consists of 4 class and 1 driver class. Those are `HotDrink` class, `ColdDrink` class, `OrderList` class, `OrderQueue` class, and `MainMenu` jframe driver class. `Hot Drink` class has 2 private attributes such as `sugar (double)` and `quantity (int)`, 1 constructor that inherit from `Drink` class, 4 overriding setter and getter methods that implement from `SugarPrefer` interface and `Quantity` interface such as `setsugar(i: double): void`, `getsugar(): double`, `setquantity(q: int): void` and `getquantity(): int`, and 1 method such `getTotal(): double` to give total price based on size and quantity. `ColdDrink` class has 2 private attributes such as `sugar (double)`, `ice (String)`, `quantity (int)`, `topname (String)`, and `topprice (Double)`, 1 constructor that inherit from `Drink` class, 10 overriding methods that implement from `SugarPrefer` interface, `ColdPrefer` interface, `Quantity` interface, and `Topping` interface such as `setsugar(i: double): void`, `getsugar(): double`, `setice(i: String): void`, `getice(): String`, `setquantity(q: int): void`, `getquantity(): int`, `ddtopping(t: String): void`, `addtopping(t: String, p: double): void`, `getTopping(): String`, and `getTopPrice(): double`, 1 method such as `getTotal()` that give total price based on size, topping, and quantity.

Next, `OrderList` class has 5 private attributes such as `d (Drink)`, `hd (HotDrink)`, `cd (ColdDrink)`, `isPaid (boolean)`, and `ol (ArrayList)` and 10 methods such as `addOrder(d: Drink): void` to add drink object into order list, `deleteOrder(d: Drink): Boolean` to delete drink object from order list, `clearOrder(): void` to delete all drink objects from order list, `getList(): ArrayList` to return the list itself, `isEmptyOrder(): boolean` to give statement that order list is empty or not, `read(f: String): void` to read objects from file into order list, `write(f: String): void` and `write(f: String, n: int): void` to write objects from order list into file, `setPaid(): void` and `isPaid(): boolean` are setter and getter to private attribute `paid`. `OrderQueue` class has 2 private attributes such as `q (Queue)` and `number (int)` and 6 methods such as `push(n: int): void` to push back queue number into queue list, `pop(): void` to pop front of queue list, `top(): number` to get front of queue list, `getNumber(): int` to give queue number, `isEmptyQueue(): Boolean` to give statement that queue list is empty or not, and `numberincrement(): void` to increase queue number.

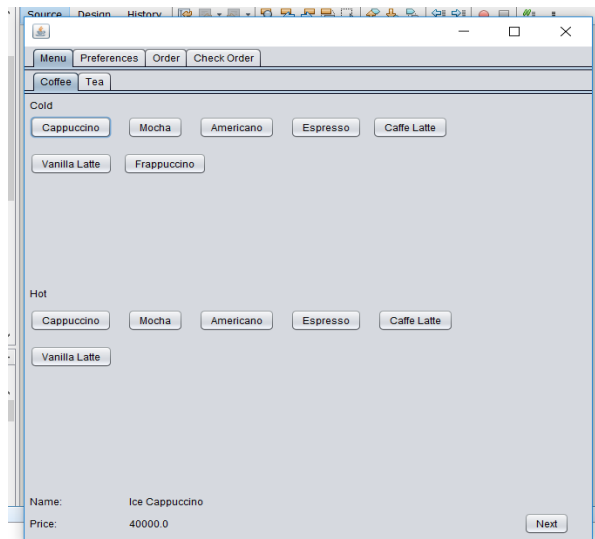
Last, `MainMenu` jframe driver class has 5 private attributes such as `d (Drink)`, `cd (ColdDrink)`, `hd (HotDrink)`, `olist (OrderList)`, and `oq (OrderQueue)` and 8 methods such as `main(): void` to start the program, `AddAllRowTable(): void` to add all drink objects from order list into order or check table, `AddtoRowTable(): void` to add a drink objects to order table, `clearTable(): void` to clear all drink objects from order or check table, 3 reset form methods (`ResetColdCoffeeForm`, `ResetColdTeaForm`, and `ResetHotDrinkForm`) to reset form after adding drink object or entering into itself, and `GetPriceFromTable(): double` to give total price from price row of order table.

3. Design



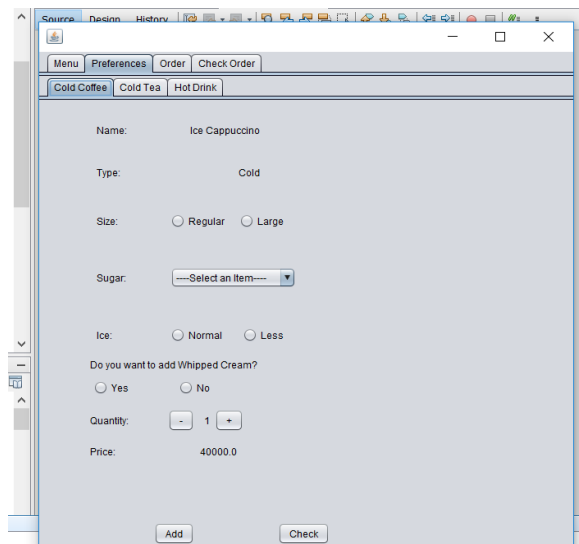


4. Result



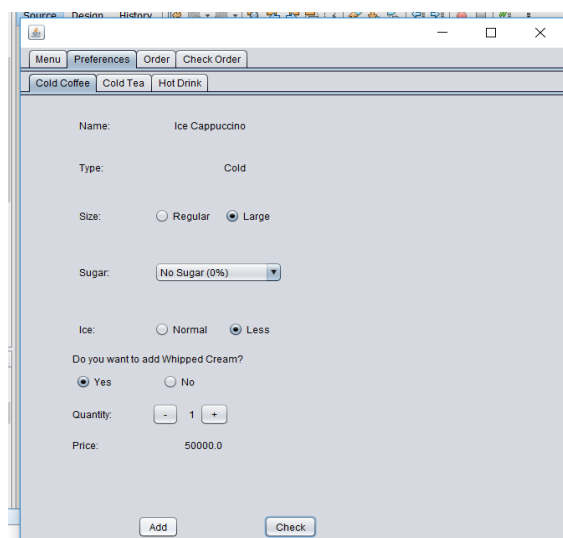
The Coffee Menu Panel is a software window with a title bar containing 'Source', 'Design', 'History', and standard window controls. It features four tabs: 'Menu', 'Preferences', 'Order', and 'Check Order'. The 'Menu' tab is active, showing a sub-tabbed interface for 'Coffee' and 'Tea'. Under the 'Coffee' sub-tab, there are two sections: 'Cold' and 'Hot'. Each section contains buttons for 'Cappuccino', 'Mocha', 'Americano', 'Espresso', 'Caffe Latte', and 'Vanilla Latte'. At the bottom, a summary shows 'Name: Ice Cappuccino' and 'Price: 40000.0', with a 'Next' button on the right.

Coffee Menu Panel



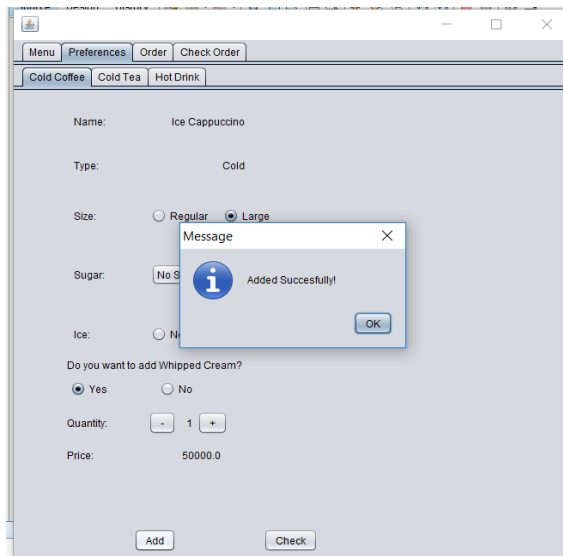
The Cold Coffee Preferences Panel is a software window with the same title bar and tabs as the menu panel. The 'Preferences' tab is active, showing sub-tabs for 'Cold Coffee', 'Cold Tea', and 'Hot Drink'. The 'Cold Coffee' sub-tab is selected. It displays the current selection: 'Name: Ice Cappuccino', 'Type: Cold', 'Size: Regular' (selected), 'Sugar: --Select an Item--', 'Ice: Normal' (selected), 'Do you want to add Whipped Cream? No' (selected), 'Quantity: 1', and 'Price: 40000.0'. 'Add' and 'Check' buttons are at the bottom.

Cold Coffee Preferences Panel



This panel shows the state after clicking the 'Check' button. The 'Cold Coffee' sub-tab remains active. The 'Size' is now 'Large' (selected), 'Sugar' is 'No Sugar (0%)', and 'Ice' is 'Less' (selected). The 'Do you want to add Whipped Cream?' option is now 'Yes' (selected). The 'Quantity' remains '1', but the 'Price' has updated to '50000.0'. The 'Add' and 'Check' buttons are still present at the bottom.

Cold Coffee Preferences Panel after clicking
“Check” button

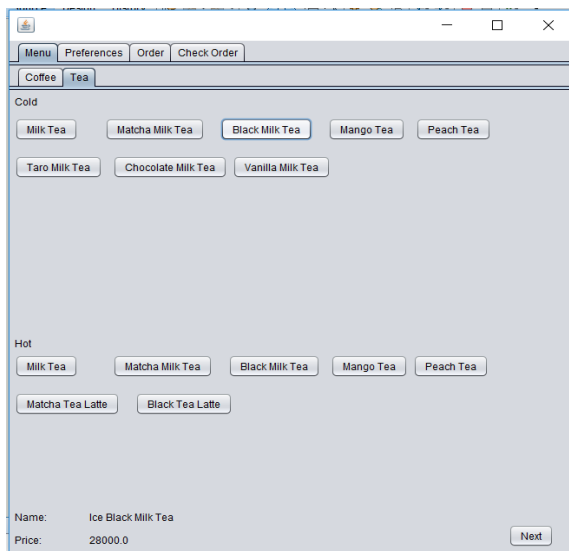


The image shows a software window titled "Menu" with tabs for "Menu", "Preferences", "Order", and "Check Order". The "Preferences" tab is active, and the "Cold Coffee" sub-tab is selected. The form displays the following details for an "Ice Cappuccino":

- Name: Ice Cappuccino
- Type: Cold
- Size: ☐ Regular ☒ Large
- Sugar:
- Ice: ☐ No Ice ☒ Ice
- Do you want to add Whipped Cream?: ☒ Yes ☐ No
- Quantity: (with minus and plus buttons)
- Price: 50000.0

At the bottom are "Add" and "Check" buttons. A modal "Message" box is overlaid on the form, displaying an information icon and the text "Added Successfully!" with an "OK" button.

Cold Coffee Preferences Panel after clicking
“Add” button



The image shows a software window titled "Menu" with tabs for "Menu", "Preferences", "Order", and "Check Order". The "Menu" tab is active, and the "Tea" sub-tab is selected. The form displays two sections of tea options:

Cold

- Milk Tea
- Matcha Milk Tea
- Black Milk Tea
- Mango Tea
- Peach Tea
- Taro Milk Tea
- Chocolate Milk Tea
- Vanilla Milk Tea

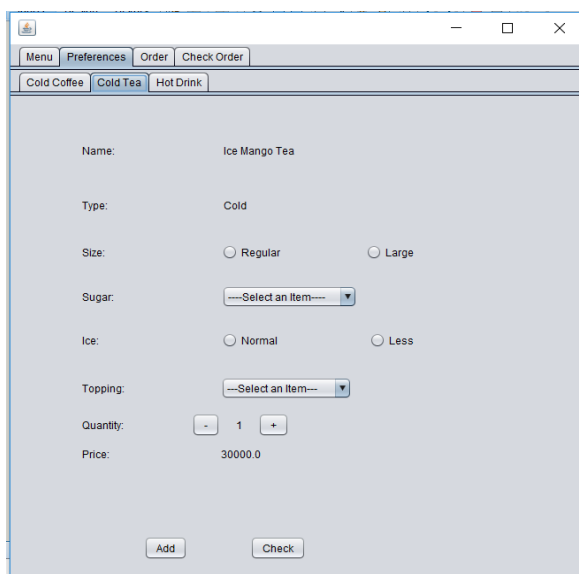
Hot

- Milk Tea
- Matcha Milk Tea
- Black Milk Tea
- Mango Tea
- Peach Tea
- Matcha Tea Latte
- Black Tea Latte

At the bottom, the form shows:

- Name: Ice Black Milk Tea
- Price: 28000.0
- Next button

Tea Menu Panel

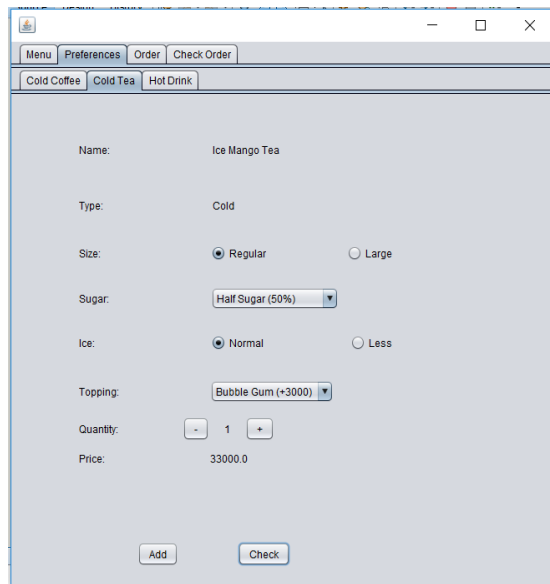


The image shows a software window titled "Menu" with tabs for "Menu", "Preferences", "Order", and "Check Order". The "Preferences" tab is active, and the "Cold Tea" sub-tab is selected. The form displays the following details for an "Ice Mango Tea":

- Name: Ice Mango Tea
- Type: Cold
- Size: ☐ Regular ☐ Large
- Sugar:
- Ice: ☐ Normal ☐ Less
- Topping:
- Quantity: (with minus and plus buttons)
- Price: 30000.0

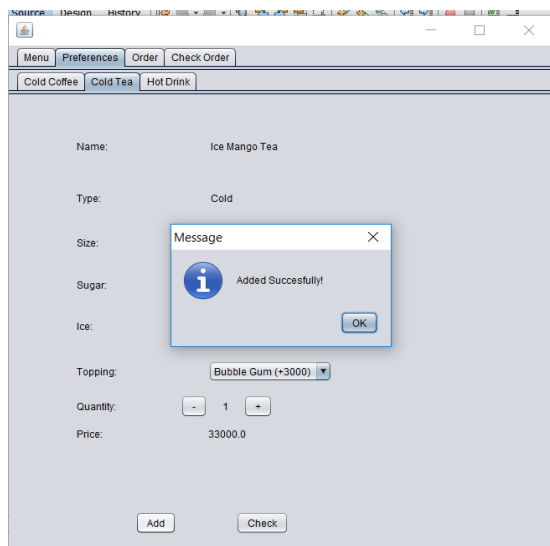
At the bottom are "Add" and "Check" buttons.

Cold Tea Preferences Panel



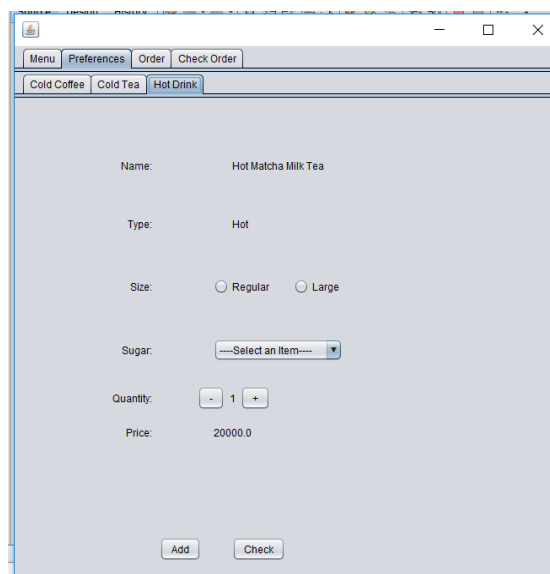
A screenshot of a web application window titled "Menu | Preferences | Order | Check Order". The "Cold Tea" tab is selected. The form displays the following details for "Ice Mango Tea":
Name: Ice Mango Tea
Type: Cold
Size: ☒ Regular ☐ Large
Sugar:
Ice: ☒ Normal ☐ Less
Topping:
Quantity:
Price: 33000.0
At the bottom are "Add" and "Check" buttons.

Cold Tea Preferences Panel after clicking
“Check” button



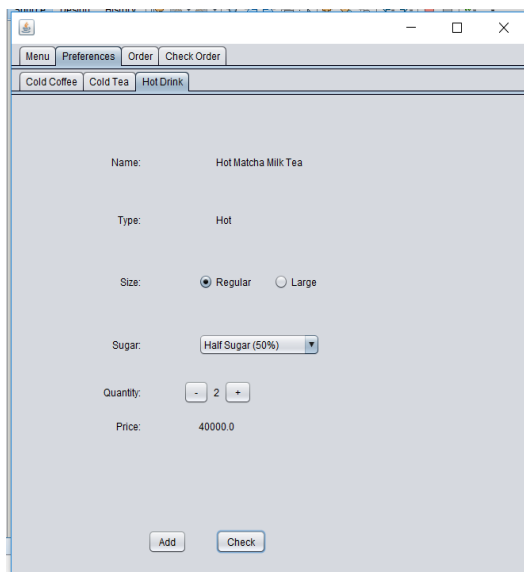
A screenshot of the same web application window, but with a modal "Message" dialog box displayed in the center. The dialog contains an information icon and the text "Added Successfully!". The "Add" button is highlighted. The background form shows the same details as the previous image.

Cold Tea Preferences Panel after clicking
“Add” button



A screenshot of the web application window with the "Hot Drink" tab selected. The form displays the following details for "Hot Matcha Milk Tea":
Name: Hot Matcha Milk Tea
Type: Hot
Size: ☐ Regular ☐ Large
Sugar:
Quantity:
Price: 20000.0
At the bottom are "Add" and "Check" buttons.

Hot Drink Preferences Panel



Menu Preferences Order Check Order

Cold Coffee Cold Tea Hot Drink

Name: Hot Matcha Milk Tea

Type: Hot

Size: ☒ Regular ☐ Large

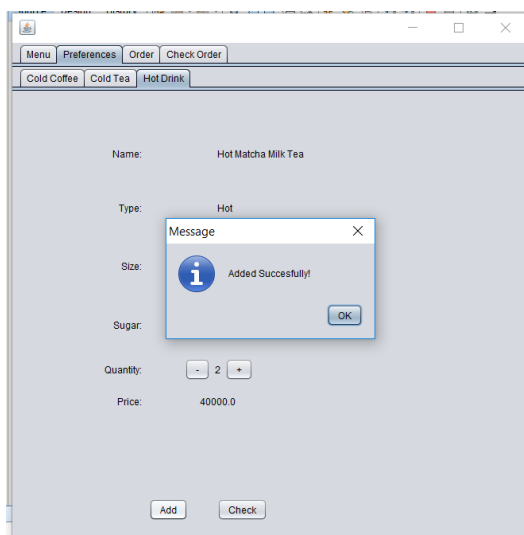
Sugar: Half Sugar (50%)

Quantity: - 2 +

Price: 40000.0

Add Check

Hot Drink Preferences Panel after clicking “Check” button



Menu Preferences Order Check Order

Cold Coffee Cold Tea Hot Drink

Name: Hot Matcha Milk Tea

Type: Hot

Size:

Sugar:

Quantity: - 2 +

Price: 40000.0

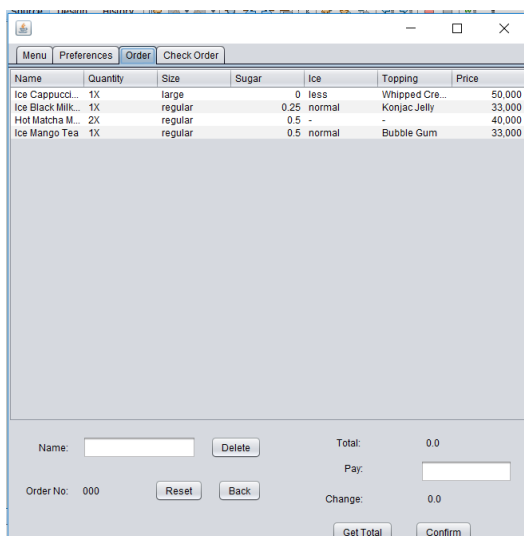
Add Check

Message

Added Successfully!

OK

Hot Drink Preferences Panel after clicking “Add” button



Menu Preferences Order Check Order

Name	Quantity	Size	Sugar	Ice	Topping	Price
Ice Cappucci...	1X	large	0	less	Whipped Cre...	50,000
Ice Black Milk...	1X	regular	0.25	normal	Konjac Jelly	33,000
Hot Matcha M...	2X	regular	0.5	-	-	40,000
Ice Mango Tea	1X	regular	0.5	normal	Bubble Gum	33,000

Name: Delete Total: 0.0

Order No: 000 Reset Back Pay:

Change: 0.0

Get Total Confirm

Order List Panel after clicking “Ok” button on “Added Successfully!” notification

The screenshot shows a software window titled "Menu" with tabs for "Menu", "Preferences", "Order", and "Check Order". The "Order" tab is active, displaying a table with columns: Name, Quantity, Size, Sugar, Ice, Topping, and Price. The table contains four rows of items: Ice Cappuccino (1X, large, 0 less, Whipped Cr., 50,000), Ice Black Milk Tea (1X, regular, 0.25 normal, Konjac Jelly, 33,000), Hot Matcha Milk T... (2X, regular, 0.5 -, 40,000), and Ice Mango Tea (1X, regular, 0.5 normal, Bubble Gum, 33,000). Below the table, there is a "Name:" label followed by a text input field containing "Ice Black Milk Tea" and a "Delete" button. To the right, there are labels for "Total:" (0.0), "Pay:" (empty field), "Order No:" (000), "Reset", "Back", "Change:" (0.0), "Get Total", and "Confirm" buttons.

Name	Quantity	Size	Sugar	Ice	Topping	Price
Ice Cappuccino	1X	large	0	less	Whipped Cr...	50,000
Ice Black Milk Tea	1X	regular	0.25	normal	Konjac Jelly	33,000
Hot Matcha Milk T...	2X	regular	0.5	-	-	40,000
Ice Mango Tea	1X	regular	0.5	normal	Bubble Gum	33,000

Name: Delete

Total: 0.0

Pay:

Order No: 000 Reset Back

Change: 0.0

Get Total Confirm

Order List Panel after filling order name text field

This screenshot shows the same software window as the previous one, but with a "Message" dialog box displayed in the center. The dialog box has a title bar "Message" and a close button "X". It contains an information icon (i) and the text "Deleted Successfully!". Below the text is an "OK" button. The background window is slightly dimmed, showing the same table and form elements as before.

Message

Deleted Successfully!

OK

Order List Panel after clicking "Delete" button

The screenshot shows the software window after the "Message" dialog box has been closed. The "Name:" text input field is now empty, and the "Delete" button is still present. All other elements, including the table and the bottom form, remain the same as in the previous screenshots.

Name: Delete

Total: 0.0

Pay:

Order No: 000 Reset Back

Change: 0.0

Get Total Confirm

Order List Panel after clicking "Ok" button on "Deleted Successfully" notification.

Name	Quantity	Size	Sugar	Ice	Topping	Price
Ice Cappuccino	1X	large	0	less	Whipped Cr...	50,000
Hot Matcha Milk T...	2X	regular	0.5	-	-	40,000
Ice Mango Tea	1X	regular	0.5	normal	Bubble Gum	33,000

Name: Delete

Total: 123000.0

Pay:

Order No: 1 Reset Back

Change: 0.0

Get Total Confirm

Order List Panel after clicking “Get Total” button

Name	Quantity	Size	Sugar	Ice	Topping	Price
Ice Cappuccino	1X	large	0	less	Whipped Cr...	50,000
Hot Matcha Milk T...	2X	regular	0.5	-	-	40,000
Ice Mango Tea	1X	regular	0.5	normal	Bubble Gum	33,000

Name: Delete

Total: 123000.0

Pay:


Order No: 1 Reset Back

Change: 0.0

Get Total Confirm

Order List Panel after filling payment text field

Message

 Payment is complete!

OK

Name	Quantity	Size	Sugar	Ice	Topping	Price
Ice Cappuccino	1X	large	0	less	Whipped Cr...	50,000
Hot Matcha Milk T...	2X	regular	0.5	-	-	40,000
Ice Mango Tea	1X	regular	0.5	normal	Bubble Gum	33,000

Name: Delete

Total: 123000.0

Pay:

Order No: 1 Reset Back

Change: 0.0

Get Total Confirm

Order List Panel after clicking “Confirm” button

button

The screenshot shows a window titled "Order List Panel" with tabs for "Menu", "Preferences", "Order", and "Check Order". The "Order" tab is active, displaying a table with the following data:

Name	Quantity	Size	Sugar	Ice	Topping	Price
Ice Cappuccino	1X	large		0 less	Whipped Cr...	50,000
Hot Matcha Milk T...	2X	regular	0.5 -			40,000
Ice Mango Tea	1X	regular	0.5 normal		Bubble Gum	33,000

Below the table, there are input fields and buttons:

- Name: Delete
- Total: 123000.0
- Pay: 125000
- Order No: 1 Reset Back
- Change: 2000.0
- Get Total Confirm

Order List Panel after clicking “Ok” on “Payment is Complete” notification

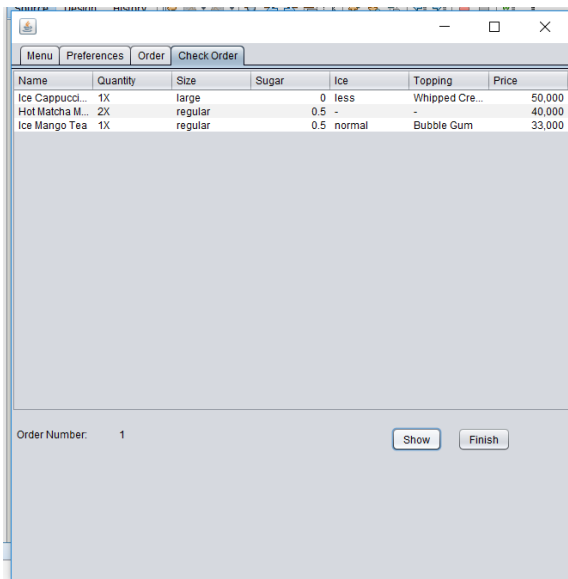
The screenshot shows the same "Order List Panel" window, but with a "Message" dialog box open in the center. The dialog box contains an information icon and the text "Order Queue has been added", with an "OK" button at the bottom.

Order List Panel after clicking “Reset” button

The screenshot shows a window titled "Check Order Panel" with tabs for "Menu", "Preferences", "Order", and "Check Order". The "Check Order" tab is active, displaying a form with the following fields and buttons:

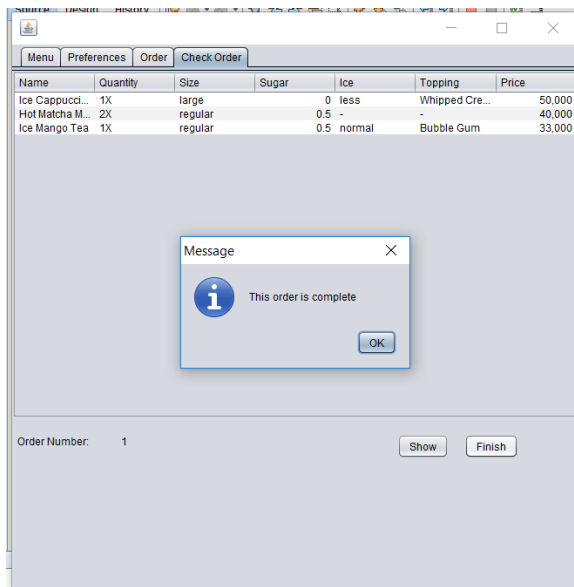
- Order Number: 000
- Show Finish

Check Order Panel



Check Order Panel after clicking “Show”

button



Check Order Panel after clicking “Finish”

button

5. References

Anonymous, 2016. Retrieved June 6, 2019, from:

<https://1bestcsharp.blogspot.com/2016/03/java-populate-jtable-from-arraylist.html>

Anonymous, 2011. Retrieved June 7, 2019, from:

<https://stackoverflow.com/questions/4577792/how-to-clear-jtable/4578501>

Dickshan, Sulanjala, 2016. How To get Sum From JTable all rows Java Swing Netbeans. Retrieved June 19, 2019, from:

<https://www.youtube.com/watch?v=U0JFpBhdK4M>

Anonymous, 2012. Retrieved June 20, 2019, from:

<https://stackoverflow.com/questions/12048864/resetting-the-value-of-a-jcombobox>

Abode, Intact, 2017. How to Get the JCombobox Value in and java Swing `getSelectedItem`. Retrieved June 2, 2019, from:

<https://www.youtube.com/watch?v=hSghISXr7b8>

Anonymous, 2013. Retrieved June 3, 2019, from:

<https://stackoverflow.com/questions/15080256/clearing-a-group-of-radio-buttons-in-java>