

Vending Machine

Ass2B00107874

[David Mulhall] | [Software development] | [06/12/2017]

# Objective

The main aim of the assignment is to produce a Java program that can emulate and control the vending machine

The user should be allowed select from for options when an option is chosen the appropriate message should be displayed to the screen when option 5 is chosen the program will exit. Upon exiting the program a receipt will be shown to the user clearly displaying all drinks sold during this transaction if any other up as chosen an error message will be displayed to the user a minimum of at least one user defined method should be used in this program.

# Pseudocode

* Initialise variables
* Begin init method
* Create labels and text fields and add to program
* Create paint method
* Create event listener
* Paint string to screen
* Create switch statement
* Take user input and use in switch statement
* On switch statement increase chosen drink variable by 1
* If accept and pay is chosen then call print receipt method
* Create print receipt method
* Display final text to screen

# Source code

import java.applet.Applet;

import java.awt.\*;

import java.awt.Font;

import javax.swing.\*;

import java.awt.event.\*;

import java.text.DecimalFormat;

public class VendingMachine extends Applet implements ActionListener

{

//Declare labels

Label lbl1,lbl2,lbl3,lbl4;

// Declare TextField

TextField select;

//Declare Font

Font myFont;

//Declare Variables

String choice;

int coffee=0,tea=0,soup=0,water=0,page=0;

Double total=0.00;

public void init()

{

//create label and TextFields

Font myFont = new Font("monospaced", Font.BOLD, 50);

lbl1 = new Label("Welcome to");

lbl2 = new Label("Nuka-Cola");

lbl3 = new Label("Please make your selection: ");

lbl4 = new Label((space(29))+"");

select = new TextField ();

//add Buttons and Labels

add(lbl1);

add(lbl2);

add(lbl3);

add(select);

add(lbl4);

//other attributes

lbl1.setFont(myFont);

lbl2.setFont(myFont);

select.addActionListener( this );

}

//start paint method

public void paint(Graphics g)

{

//Define decimal formatting

DecimalFormat df = new DecimalFormat("€ 0.00")

// start page choice if statement

if(page==0)

{//first page

g.drawString("# 1 Coffee ", 110, 225);

g.drawString("# 2 Tea ", 205, 225);

g.drawString("# 3 Soup ", 110, 250);

g.drawString("# 4 Water ", 205, 250);

g.drawString("# 5 Finish and Pay", 135, 285);

showStatus("SubTotal: "+(df.format(total))); }

else

{//reciept page

g.drawString("You ordered :", 70, 150);

g.drawString(coffee+" Coffee's @ €2.00 ", 150, 150);

g.drawString(tea+" Tea's @ €2.00 ", 150, 175);

g.drawString(soup+" Soup's @ €2.00 ", 150, 200);

g.drawString(water+" Water's @ €1.50 ", 150, 225);

g.drawString("Your total is "+(df.format(total)), 150, 275);

showStatus("Thank you, Have a nice day!");

lbl3.setVisible(false);

lbl4.setVisible(false);

}

}

//start event handler

public void actionPerformed( ActionEvent e )

{

//set choice to be text entered in textField

choice = select.getText();

//start switch statement

switch(choice)

{

case "1": coffee++; select.setText("");lbl4.setText("You have chosen Coffee");timer(1);repaint();

break;

case "2": tea++; select.setText("");lbl4.setText("You have chosen Tea");timer(1);repaint();

break;

case "3": soup++; select.setText("");lbl4.setText("You have chosen Soup");timer(1);repaint();

break;

case "4": water++; select.setText("");lbl4.setText("You have chosen Water");timer(1);repaint();

break;

case "5": select.setVisible(false);page=1;repaint();

break;

default: select.setText("");lbl4.setText("Please make a valid selection");timer(1);repaint();

break;

}

//call total to be worked out by totalDue method.

totalDue(coffee,tea,soup,water);

}

//timer uses brute force number crunching to force the cpu

//to stall for a moment. it is used to clear the message box

//after it tells the user what drink they have chosen

public void timer(int par1)

{

for (int i=(par1\*10000);i>=0;i--)

{

System.out.print(Integer.toString(i));

if (i<=5)

{

lbl4.setText("");

}

}

}

//generates a string of empty spaces from a passed parameter

//and returns the string for text positioning purposes

public String space(int par1)

{

String blank = "";

String add =" ";

for (int i=0;i<par1;i++)

{

blank+=(add);

}

return blank;

}

public Double totalDue(int par1,int par2,int par3,int par4)

{

//calculate total

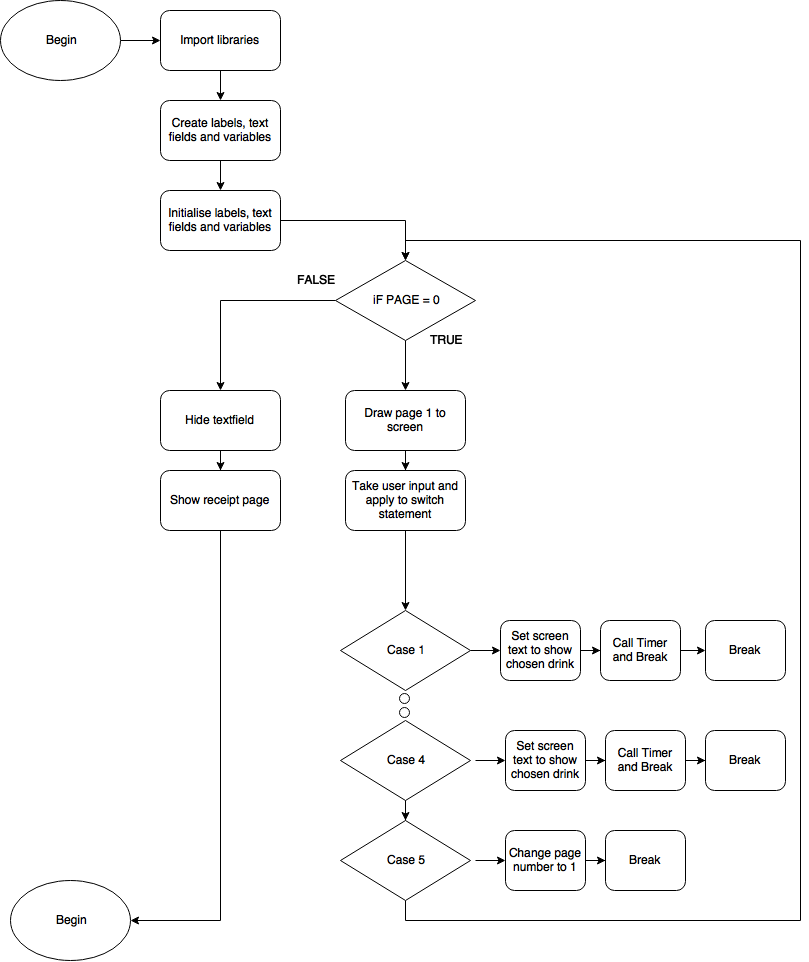
total = (par1\*2.00)+(par2\*2.00)+(par3\*2.00)+(par4\*1.50);

return(total);

}

}

# FlowChart



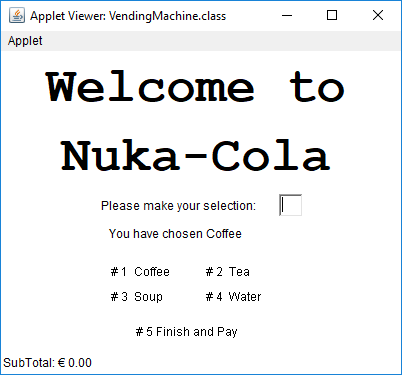
# Test Cases

Listed below is a series of tests that will be ran in order to verify that all aspects of the program are working correctly and that any potential errors or failures are identified and rectified. Aspects tested will be :

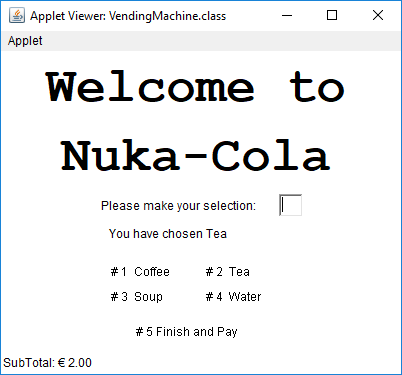
1. Check that selection 1 to 4 are working
2. Check that selection 5 is working
3. Check that total addition is working correctly
4. Check that the system can handle inputs that don’t fall under 1-5

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Description** | **Expected**  **result** | **Actual**  **result** |
| **1** | Check that selection 1 to 4 are working | Numbers 1 to 4 should show up a confirmation message then add 1 to the chosen variable | *Acknowledgement is shown and variable is increased*  *\*Screenshot 1 + 2* |
|  |  |  |  |
| **2** | Check that selection 5 is working | Acknowledge input and show receipt page | *Receipt page shows correctly*  *\*Screenshot 3* |
|  |  |  |  |
| **3** | Check that total addition is working correctly | 1 coffee, 1 tea, 1 soup and 1 water should cost EUR 7.50 | *Addition works correctly*  *\*Screenshot 4* |
|  |  |  |  |
| **4** | Check that the system can handle inputs that don’t fall under 1-5 | Zero and letters will cause a brief error message to appear and then the screen will reload | Error shown as expected  *\*Screenshot 4* |

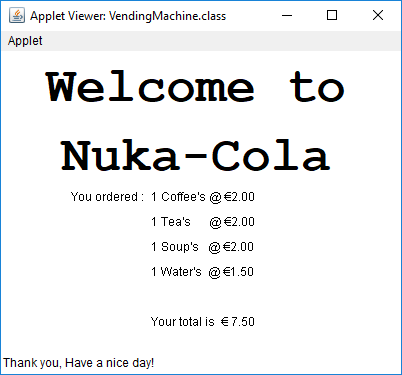
# ScreenShot 1



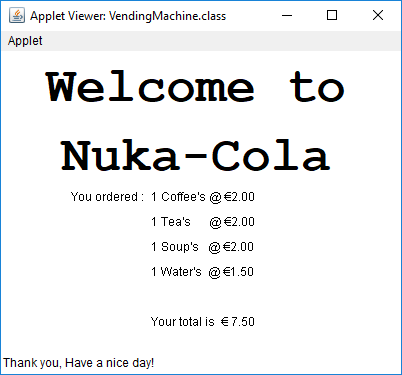
# ScreenShot 2



# ScreenShot 3



# ScreenShot 4



# ScreenShot 5

