CSC131 Fall 2018 Santiago Bermudez

Lab: Gaining Experience with GUI Construction

**Instructions:** Answer as many of the following questions as you can during the lab period. If you are unable to complete the assignment during the lab period it is strongly recommended that you complete it on your own.

You may work on this assignment alone or in a group.

**Deliverables:** Submit 1) screenshots of the interfaces you create and 2) a zip file including all java source code in Canvas.

**Getting Ready:** Before going any further, you should:

1. Setup your development environment.
2. Download the following files from the lab instruction page on Canvas: [CashMachine.java](https://users.cs.jmu.edu/bernstdh/web/common/labs/experience_gui-construction/CashMachine.java)

[PINPadWindow.java](https://users.cs.jmu.edu/bernstdh/web/common/labs/experience_gui-construction/PINPadWindow.java) [NumberPad.java](https://users.cs.jmu.edu/bernstdh/web/common/labs/experience_gui-construction/NumberPad.java)

to an appropriate directory/folder. (In most browsers/OSs, the easiest way to do this is by right-clicking/control-clicking on each of the links above.)

1. ***Working with Windows:*** This part of the lab will give you some experience with windows.

1.

Execute CashMachine. What happened?

A window popped up. It is labeled by “ATM” at the top and has title bar controls. This window is currently empty.

2.

Click on the icon that closes the window (which will vary with the operating system you are using). What happened? (Be careful!)

When I clicked on the “X” icon, it closed the window, but the application did not terminate.

* 1. Terminate CashMachine.
  2. Add the following statement to the end of the constructor in the PINPadWindow class.

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

5.

Execute CashMachine and again click on the icon that closes the window (which will vary with the operating system you are using). What happened now?

This time, the window closed and the application successfully terminated without me needing to terminate it manually.

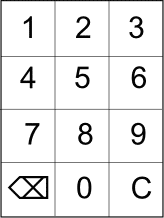
1. ***Layout:*** This part of the lab will give you some experience constructing GUI components and working with layout managers.

1.

Suppose you needed to layout a Container in a table/matrix that contains four rows and three columns. What layout manager would you use?

new GridLayout(4, 3);

2.



Complete the setupLayout() method in the NumberPad class. Your implementation must contain 12 JButton objects and be consistent with the following wireframe.

**Del**

What code did you add?

**private** **void** setupLayout()

{

// Setup and layout this NumberPad

JButton button;

setLayout(**new** GridLayout(4, 3));

**for**(**int** i = 1; i <= 9; i++) {

button = **new** JButton(String.*format*("%1d", i));

add(button);

}

button = **new** JButton("Del");

add(button);

button = **new** JButton("0");

add(button);

button = **new** JButton("C");

add(button);

}

3.

Your implementation may include duplicate code. If so, correct your implementation by adding one or more private methods (and, perhaps, a "constant"). What does your code look like now?

**private** **void** addButton(String text) {

JButton button;

button = **new** JButton(text);

add(button);

}//\*PLEASE NOTE THAT I COULDN”T RESIZE THE BOX SO THE CODE IS SQUISHED A BIT

**private** **void** setupLayout(){

setLayout(**new** GridLayout(4, 3));

**for**(**int** i = 1; i <= 9; i++) { addButton(String.*format*("%1d", i));}

addButton("Del");

addButton("0");

addButton("C");

}

4.

Modify the setupLayout() method **in the PINPadWindow** class so that it now constructs a NumberPad and adds it to the content pane. What code is in the setupLayout() method now?

**private** **void** setupLayout(){

Container contentPane;

NumberPad numberPad;

setSize(300, 300);

setTitle("ATM");

contentPane = getContentPane();

numberPad = **new** NumberPad();

contentPane.add(numberPad);

}

5.

Execute CashMachine. How big is the window and how big are the buttons?

Based on our code, the window is 300x300 and the buttons are 100x75(\*I got this by dividing 300 by 3 and 4). Also, 300x300 is (Horizontal length x Vertical length).

6. Terminate CashMachine.

7.

|  |  |
| --- | --- |
| Add the following to the bottom of the constructor in the PINPadWindow class. | |
|  | pack(); |
| Execute CashMachine. How big is the window now?  The window is just barely big enough to hold the buttons now. | |

8.

Resize the window. What happens and why?

As we resize the window, the buttons resize to accommodate. They get larger horizontally and/or vertically to fill up the window. This can be because that’s how GridLayout works.

9.

|  |  |
| --- | --- |
| Add the following to the bottom of the constructor in the PINPadWindow class. | |
|  | setResizable(false); |
| Execute CashMachine. Can you re-size the window?  No, we cannot. | |

1. ***Specializing GUI Components:*** This part of the lab will give you some experience adding capabilities to GUI components using specialization.



Create a Display class that specializes the JLabel class. The default constructor must call the

single-parameter constuctor in the parent class passing it " " and then call the setBorder()

method passing it an etched border. What code is in the class?

**import** javax.swing.\*;

public class Display extends JLabel {

**public** Display() {

**super**(" ");

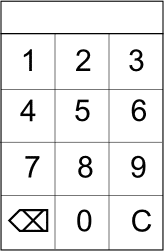
setBorder(BorderFactory.*createEtchedBorder*());

}

}

1.

***4.More Layout:*** This part of the lab will give you more experience with layout. 1.



Modify the setupLayout() method in the PINPadWindow class so that it adds a Display

above the NumberPad in a fashion that is consistent with the following wireframe.

**Del**

What code is in this method now? (Note: Remember to construct an appropriate layout manager and pass it to setLayout().

**private** **void** setupLayout()

{

Container contentPane;

Display display;

NumberPad numberPad;

setSize(300, 300);

setTitle("ATM");

contentPane = getContentPane();

contentPane.setLayout(**new** BorderLayout());

display = **new** Display();

contentPane.add(display, BorderLayout.***NORTH***);

numberPad = **new** NumberPad();

contentPane.add(numberPad, BorderLayout.***CENTER***);

// Layout this PINPadWindow

}

1. ***Event-Driven Programming:*** This part of the lab will give you some experience with event-driven programming.



Modify the Display class so that it now realizes the ActionListener interface. Specifically, it should respond to ActionEvent objects that have an action command of "C" by clearing its contents (i.e., by setting its text to " ") and any other ActionEvent objects by appending the action command to its current contents.

What code is in this class now?

**import** javax.swing.\*;

**import** java.awt.\*;

**import** java.awt.event.\*;

**public** **class** Display **extends** JLabel **implements** ActionListener {

**public** **static** **final** String ***CLEAR*** = "C";

**public** Display() {

**super**(" ");

setBorder(BorderFactory.*createEtchedBorder*());

}

**public** **void** actionPerformed(ActionEvent ae) {

String ac;

ac = ae.getActionCommand();

**if**(ac.equals(***CLEAR***)) setText(" ");

**else** setText(getText()+ac);

}

}

1.

2.

Modify the constructor in the NumberPad class so that it is now passed an ActionListener object that it stores in a private attribute named listener, before calling setupLayout(). What code is in the constructor now?

**public** NumberPad(ActionListener listener){

**super**();

**this**.listener = listener;

setupLayout();

}

3.

Modify the addButton() method in the NumberPad class so that it makes listener an

ActionListener on the button it is adding. What code did you add?

button.addActionListener(listener);

4.

Modify the setupLayout() method in the PINPadWindow class so that the Display is now an ActionListener on the NumberPad.

What code did you change?

numberPad = **new** NumberPad();

to

numberPad = **new** NumberPad(display);

5.

Execute CashMachine. What happens when you click on the various buttons?

When you click the various numbers on the keypad, they get added to the left of the display in the order that they were clicked. The “C” or clear button clears all numbers on the display. The “Del” button is the only thing that does not work as expected as instead of deleting the rightmost number, it adds Del onto the display.

6.

Modify the Display class so that the text is aligned to the right. What change did you make?

**super**(" ");

to

**super**(" ", SwingConstants.***RIGHT***);

7.

Now, modify the Display so that:

1. The "Del" button works as expected.
2. When the Display is empty it displays the text "Enter your PIN".
3. When the Display is not empty it shows (an appropriate number of) asterisks rather than the PIN.

What code is in your Display class now?

**import** javax.swing.\*;

**import** java.awt.\*;

**import** java.awt.event.\*;

**public** **class** Display **extends** JLabel **implements** ActionListener {

**private** String contents;

**public** **static** **final** String ***CLEAR*** = "C";

**public** **static** **final** String ***DELETE*** = "Del";

**public** Display() {

//super(" ");

**super**(" ", SwingConstants.***RIGHT***);

setBorder(BorderFactory.*createEtchedBorder*());

contents = "";

updateDisplay();

}

**public** **void** actionPerformed(ActionEvent ae) {

String ac;

ac = ae.getActionCommand();

**if**(ac.equals(***CLEAR***)) {

contents = "";

setText("Enter your PIN");

}**else** **if**(ac.equals(***DELETE***)) {

**if**(!contents.equals(""))

contents = contents.substring(0, contents.length()-1);

}**else** {

contents += ac;

}

updateDisplay();

}

**private** **void** updateDisplay() {

**if**(contents.equals("")) setText("Enter your PIN");

**else** {

String asterisks = "";

**for**(**int** i = 0; i < contents.length(); i++) {

asterisks += "\*";

}

setText(asterisks);

}

}

}

8.

Now, modify the Display so that the text is gray when "Enter your PIN" is displayed and red when the asterisks are displayed. What code did you change?

**private** **void** updateDisplay() {

**if**(contents.equals("")) setText("Enter your PIN");

**else** {

String asterisks = "";

**for**(**int** i = 0; i < contents.length(); i++) {

asterisks += "\*";

}

setText(asterisks);

}

}

to

**private** **void** updateDisplay() {

**if**(contents.equals("")) {

setForeground(Color.***GRAY***);

setText("Enter your PIN");

}**else** {

String asterisks = "";

**for**(**int** i = 0; i < contents.length(); i++) {

asterisks += "\*";

}

setForeground(Color.***RED***);

setText(asterisks);

}

}