**CIS 162 Lab 10**

**Graphical User Interfaces (GUI)**

**Objectives**

After completing this lab, you should be able to:

* *Position* buttons and text fields in a JFrame
* *write* an action listener
* *describe* the advantage of separating the model and the GUI (view)
* *describe* how the user controls program actions by clicking on a button

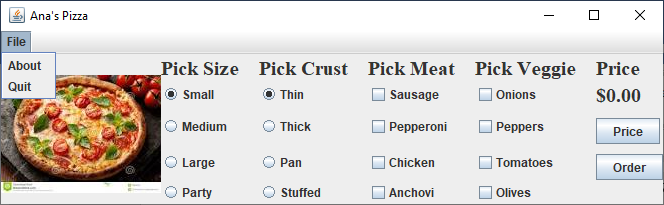
**Model-View Design Pattern**

A common design strategy is to divide applications into at least two parts: the user interface (View) and a back end that takes care of the data (Model). For this lab, we provide a completed Model class that stores a pizza order and is responsible for calculating the cost. You will expand an initial GUI that allows a customer to specify an order.

The Model-View pattern enforces clear responsibilities between the model and view. For example, the GUI (view) should not be responsible for maintaining any data or knowing how the data are processed. The sole responsibility of the GUI is to support user interaction with the Model.

**Sample Display**

Your GUI will eventually look similar to the following:



**Step 1: Create a New BlueJ Project**

**Step 2: Download the provided PizzaOrder class to your project**

Rather than writing your own class, we are providing a completed class for you.

Right click on the PizzaOrder.class file to download and Save As… on your computer in the folder for the newly created BlueJ project. Pay attention where it is saved and be sure to provide the correct name! You might not see the class icon within BlueJ but try closing and the opening again.

**PizzaOrder API**

These are the methods of the PizzaOrder class:

* PizzaOrder() – constructor
* void clearOrder() – reset order to small, thin crust with no meats or veggies.
* void setSize(int size) – set the pizza size. Valid parameters include: PizzaOrder.SMALL, PizzaOrder.MEDIUM, PizzaOrder.LARGE and PizzaOrder.PARTY.
* void setCrust(int crust) – set the crust style. Valid parameters include: PizzaOrder.THIN, PizzaOrder.THICK, PizzaOrder.PAN and PizzaOrder.STUFFED.
* void addMeat() – add one meat topping to the order.
* void addVeggie() – add one veggie topping to the order.
* double getPrice() – calculate and return the price.
* String getOrder() – return a text summary of the order.

**Sample Usage**

PizzaOrder theOrder = new PizzaOrder();

theOrder.setSize(PizzaOrder.LARGE);

theOrder.setCrust(PizzaOrder.PAN);

theOrder.addMeat();

double price = theOrder.getPrice();

String summary = theOrder.getOrder();

**Step 3: Modify the provided PizzaGUI class**

Rather than writing your own GUI class, we are providing code to get you started. Save the PizzaGUI.java file in the folder that BlueJ created for your project. This class should compile with no changes.

We modeled our design on examples from the book in section 10.3. Read the code and comments carefully to understand how it works. You start the program by invoking main().

**NOTE: Look for FIX ME comments in the code.**

**Instance Variables:**

**Add sizes and crusts**

* Add size options for Large and Party. You must define additional JRadioButtons and use the current code as a guide.
* Add crust options for Pan and Stuffed. You must define additional JRadioButtons and use the current code as a guide.

**Add meat and veggie options**

* Add at least two more meat options of your choice. You must define additional JCheckboxes and use the current code as a guide.
* Add at least two more veggie options of your choice. You must define additional JCheckboxes and use the current code as a guide.

**Constructor:**

**Instantiate the model object**

* Instantiate the PizzaOrder object called theOrder

theOrder = new PizzaOrder();

**setupGUI method:**

Look for the // FIX ME: comments to complete the code for this method.

**actionPerformed method:**

* Add if statements within updatePizzaOrder() to let the model object know what sizes and crust were selected.
* Add if statements within updatePizzaOrder() to let the model object know what veggies and meats were selected.
* Add an if statement to process the menu item About, to display your names within a pop-up JOptionPane message.

JOptionPane.showMessageDialog(this,"your name goes here");

* Add an if statement within actionPerformed() to display the current order. You will have to ask the PizzaOrder object for the order summary.

String summary = theOrder.getOrder();

JOptionPane.showMessageDialog(this, summary);

**Add an Image**

Code is provided to display an appropriate pizza image to the left of the frame.

* Find an appropriate “jpg” or “gif” image on the Internet and download to your BlueJ project folder.
* Update the code to reflect the filename of your image.

**What to turn in**? Use Blackboard to turn in your lab.

* Upload the PizzaGUI.java file
* Upload a screenshot of the GUI working

**Grading Criteria**

lab is worth 10 points.