```
import java.awt.BorderLayout;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.GridLayout;
import java.awt.Insets;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;
import java.text.NumberFormat;

import javax.swing.ComboBoxModel;
import javax.swing.DefaultComboBoxModel;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JSpinner;
import javax.swing.JTextField;
import javax.swing.WindowConstants;
import javax.swing.event.ChangeEvent;
import javax.swing.event.ChangeListener;
import javax.swing.SwingUtilities;

public class JavaZonFrame extends javax.swing.JFrame implements ActionListener, ItemListener
{
        private JPanel jPanel1;
        private JButton btnAdd;
        private JComboBox<String> cboItem1, cboItem2, cboItem3;
        JSpinner txtQty1, txtQty2, txtQty3;
        private JTextField txtPrice1, txtPrice2, txtPrice3;
        private JTextField txtLineTotal3, txtLineTotal2, txtLineTotal1;
        private JLabel jLabel12, jLabel11, jLabel10, jLabel2;
        private JLabel jLabel3, jLabel4, jLabel5, jLabel9;
        private JLabel jLabel8, jLabel7, jLabel6, jLabel1, jLabel13;
        private JTextField txtOrderID, txtFirstName, txtLastName, txtPhone, txtMemberShip;
        private JButton btnReset;
        private JTextField txtStreet, txtCity, txtState, txtZip;
        private JTextField txtTotal;
        private JTextField txtTax;
        private JTextField txtSubtotal;
        private JLabel lblLineTotal;
        private JLabel lblTotal;
        private JLabel lblTax;
        private JLabel lblSubtotal;
        private JButton btnClear;
        private JButton btnList;
        private JButton btnFind;
        private JButton btnDelete;
        private JavaZon jz;
        private Menu menu;

        public JavaZonFrame()
        {
                try
                {

                        setDefaultCloseOperation(WindowConstants.DISPOSE_ON_CLOSE);

                        GridLayout thisLayout = new GridLayout();

                        jz = new JavaZon();
```

```
menu = new Menu();

String menuItems[] = menu.getMenuDescriptions();

getContentPane().setLayout(thisLayout);

jPanel1 = new JPanel();
getContentPane().add(jPanel1);
GridBagLayout jPanel1Layout = new GridBagLayout();

jPanel1.setLayout(jPanel1Layout);
jPanel1.setPreferredSize(new java.awt.Dimension(499, 426));

btnAdd = new JButton();
jPanel1.add(btnAdd, gc(1, 25, 1, 1));
btnAdd.setText("Add");
btnAdd.addActionListener(this);

btnDelete = new JButton();
jPanel1.add(btnDelete, gc(2, 25, 1, 1));
btnDelete.setText("Delete");
btnDelete.addActionListener(this);

btnFind = new JButton();
jPanel1.add(btnFind, gc(3, 25, 1, 1));
btnFind.setText("Find");
btnFind.addActionListener(this);

btnList = new JButton();
jPanel1.add(btnList, gc(4, 25, 1, 1));
btnList.setText("List");
btnList.addActionListener(this);

btnClear = new JButton();
jPanel1.add(btnClear, gc(5, 25, 1, 1));
btnClear.setText("Clear");
btnClear.addActionListener(this);

txtOrderID = new JTextField(10);
jPanel1.add(txtOrderID, gc(2, 2, 1, 1));

btnReset = new JButton();
jPanel1.add(btnReset, gc(6, 25, 1, 1));
btnReset.setText("Reset");
btnReset.addActionListener(this);

txtFirstName = new JTextField(10);
jPanel1.add(txtFirstName, gc(2, 3, 2, 1));

txtLastName = new JTextField(10);
jPanel1.add(txtLastName, gc(5, 3, 2, 9));

txtMemberShip = new JTextField(10);
jPanel1.add(txtMemberShip, gc(2, 5, 2, 1));

txtPhone = new JTextField(10);
jPanel1.add(txtPhone, gc(5, 5, 2, 9));

txtStreet = new JTextField(10);
jPanel1.add(txtStreet, gc(2, 7, 2, 1));

txtCity = new JTextField(10);
jPanel1.add(txtCity, gc(2, 9, 2, 1));
```

```
txtState = new JTextField(10);
jPanel1.add(txtState, gc(4, 9, 2, 1));

txtZip = new JTextField(10);
jPanel1.add(txtZip, gc(6, 9, 2, 1));

cboItem1 = new JComboBox<String>(menuItems);
jPanel1.add(cboItem1, gc(2, 13, 1, 1));
cboItem1.setSelectedIndex(-1);
cboItem1.addItemListener(this);

cboItem2 = new JComboBox<String>(menuItems);
jPanel1.add(cboItem2, gc(2, 15, 1, 1));
cboItem2.setSelectedIndex(-1);
cboItem2.addItemListener(this);

cboItem3 = new JComboBox<String>(menuItems);
jPanel1.add(cboItem3, gc(2, 17, 1, 1));
cboItem3.setSelectedIndex(-1);
cboItem3.addItemListener(this);

txtQty1 = new JSpinner();
jPanel1.add(txtQty1, gc(3, 13, 1, 1));

txtQty2 = new JSpinner();
jPanel1.add(txtQty2, gc(3, 15, 1, 1));

txtQty3 = new JSpinner();
jPanel1.add(txtQty3, gc(3, 17, 1, 1));

txtPrice1 = new JTextField(10);
jPanel1.add(txtPrice1, gc(4, 13, 1, 1));

txtPrice2 = new JTextField(10);
jPanel1.add(txtPrice2, gc(4, 15, 1, 1));

txtPrice3 = new JTextField(10);
jPanel1.add(txtPrice3, gc(4, 17, 1, 1));

txtLineTotal1 = new JTextField(10);
jPanel1.add(txtLineTotal1, gc(5, 13, 1, 1));

txtLineTotal2 = new JTextField(10);
jPanel1.add(txtLineTotal2, gc(5, 15, 1, 1));

txtLineTotal3 = new JTextField(10);
jPanel1.add(txtLineTotal3, gc(5, 17, 1, 1));

txtSubtotal = new JTextField(10);
jPanel1.add(txtSubtotal, gc(5, 19, 1, 1));

txtTax = new JTextField(10);
jPanel1.add(txtTax, gc(5, 21, 1, 1));

txtTotal = new JTextField(10);
jPanel1.add(txtTotal, gc(5, 23, 1, 1));

jLabel1 = new JLabel();
jPanel1.add(jLabel1, gc(1, 2, 1, 1));
jLabel1.setText("Order ID:");

jLabel2 = new JLabel();
```

```
jPanel1.add(jLabel2, gc(1, 3, 1, 1));
jLabel2.setText("First Name:");

jLabel3 = new JLabel();
jPanel1.add(jLabel3, gc(4, 3, 1, 1));
jLabel3.setText("Last Name:");

jLabel13 = new JLabel();
jPanel1.add(jLabel13, gc(2, 0, 1, 1));
jLabel13.setText("Welcome to JAVAZON III");

jLabel4 = new JLabel();
jPanel1.add(jLabel4, gc(1, 5, 1, 1));
jLabel4.setText("MemberShip:");

jLabel12 = new JLabel();
jPanel1.add(jLabel12, gc(4, 5, 1, 1));
jLabel12.setText("Phone:");

jLabel5 = new JLabel();
jPanel1.add(jLabel5, gc(1, 7, 1, 1));
jLabel5.setText("Street");

jLabel6 = new JLabel();
jPanel1.add(jLabel6, gc(1, 9, 1, 1));
jLabel6.setText("City");

jLabel7 = new JLabel();
jPanel1.add(jLabel7, gc(3, 9, 1, 1));
jLabel7.setText("State:");

jLabel8 = new JLabel();
jPanel1.add(jLabel8, gc(5, 9, 1, 1));
jLabel8.setText("Zip:");

jLabel9 = new JLabel();
jPanel1.add(jLabel9, gc(2, 11, 1, 1));
jLabel9.setText("Product:");

jLabel10 = new JLabel();
jPanel1.add(jLabel10, gc(3, 11, 1, 1));
jLabel10.setText("Quantity:");

jLabel11 = new JLabel();
jPanel1.add(jLabel11, gc(4, 11, 1, 1));
jLabel11.setText("Price:");

lblLineTotal = new JLabel();
jPanel1.add(lblLineTotal, gc(5, 11, 1, 1));
lblLineTotal.setText("LineTotal:");

lblSubtotal = new JLabel();
jPanel1.add(lblSubtotal, gc(4, 19, 1, 1));
lblSubtotal.setText("Subtotal:");

lblTax = new JLabel();
jPanel1.add(lblTax, gc(4, 21, 1, 1));
lblTax.setText("Tax:");

lblTotal = new JLabel();
jPanel1.add(lblTotal, gc(4, 23, 1, 1));
lblTotal.setText("Total:");
```

```
                // TODO ADD THE REST OF THE CONTROLS
                // REFER TO HANDOUT FOR THE CONTROLS NEEDED
                // AND THE LOCATION OF EACH ITEM

                pack();
                this.setSize(650, 500);
        }
        catch (Exception e)
        {
                e.printStackTrace();
        }
}

private GridBagConstraints gc(int x, int y, int h, int w)
{

        GridBagConstraints c = new GridBagConstraints();

        c.gridx = x;
        c.gridy = y;
        c.gridheight = h;
        c.gridwidth = w;

        c.anchor = GridBagConstraints.WEST;

        c.fill = GridBagConstraints.HORIZONTAL;

        c.insets = new Insets(5, 5, 5, 5);

        return c;
}

public void actionPerformed(ActionEvent e)
{

        // based on the object that trigerred the event
        // call the appropriate method

        // Call addOrder

        if (e.getActionCommand() == "Add")
        {
                addOrder();
        }

        // Call clearOrders

        if (e.getActionCommand() == "Clear")
        {
                clearOrders();
        }

        // Call deleteOrder

        if (e.getActionCommand() == "Delete")
        {
                deleteOrder();
        }

        // Call findOrder

        if (e.getActionCommand() == "Find")
        {
                findOrder();
        }

        // call printOrders

        if (e.getActionCommand() == "Print")
        {
                printOrders();
        }

        // call resetForm

        if (e.getActionCommand() == "Reset")
        {
                resetForm();
        }

        if (e.getActionCommand() == "List")
        {
                printOrders();
        }

}

public void itemStateChanged(ItemEvent e)
{
        // example of method to display the price of the product
        // based on what item was selected

        Object o = e.getSource();
        if (o == cboItem1)
        {
                if (cboItem1.getSelectedIndex() != -1)
                {
                        txtPrice1.setText((menu.getPrice(cboItem1.getSelectedIndex()))…
                        txtQty1.setValue(1);
                }
        }

        if (o == cboItem2)
        {
                if (cboItem2.getSelectedIndex() != -1)
                {
                        txtPrice2.setText(menu.getPrice(cboItem2.getSelectedIndex()));
                        txtQty2.setValue(1);
                }
        }

        if (o == cboItem3)
        {
                if (cboItem3.getSelectedIndex() != -1)
                {
                        txtPrice3.setText((menu.getPrice(cboItem3.getSelectedIndex()))…
                        txtQty3.setValue(1);
                }
        }

        // TODO add code to display the price
        // for the other comboboxes and textfields for the line item price

}

private void addOrder()
```

```
{
        // CREATE VARIABLES TO HOLD INPUT
        // TODO
        // PASS THE INPUT TO THE VALIDATOR\
        // THE ONLY VALIDATION IN THIS PROJECT
        // IS THAT STRING VALUES ARE NOT BLANK
        // AND NUMERICAL VALUES ARE THE CORRECT TYPE
        // FOR EXAMPLE QUANTITY SHOULD BE INTEGER
        // PRICE SHOULD BE DOUBLE
        // EVERYTHING ELSE IS OK

        Validator.clearError(); // before we start processing we clear any
                                                // errors

        String orderID = Validator.getOrderID(txtOrderID.getText());
        String firstName = Validator.getFirstName(txtFirstName.getText());

        // add the other variables to hold your values from the validator in
        // here
        // see below what variables you will need to create a customer object
        // and an order object
        String lastName = Validator.getLastName(txtLastName.getText());
        String street = Validator.getStreet(txtStreet.getText());
        String city = Validator.getCity(txtCity.getText());
        String zip = Validator.getZip(txtZip.getText());
        String state = Validator.getState(txtState.getText());
        String phone = Validator.getPhone(txtPhone.getText());
        String membership = Validator.getMemberShip(txtMemberShip.getText());

        // Example of how to validate the combobox from the validator
        // After you create the validation method called getQuantity in the
        // validator
        // uncomment the Validator.getQuantity... line
        int aQty1 = 0;
        int aQty2 = 0;
        int aQty3 = 0;
        if (this.cboItem1.getSelectedIndex() != -1)
        {
                try
                {
                        txtQty1.commitEdit();
                }
                catch (java.text.ParseException e)
                {
                        Validator.setError("Invalid Quantity\n");
                }
                aQty1 = (Integer) txtQty1.getValue();
                String TheQty = String.valueOf(aQty1);
                Validator.getQty(TheQty);
        }

        if (this.cboItem2.getSelectedIndex() != -1)
        {
                try
                {
                        txtQty2.commitEdit();
                }
                catch (java.text.ParseException e)
                {
                        Validator.setError("Invalid Quantity\n");
                }
                aQty2 = (Integer) txtQty2.getValue();
                String TheQty = String.valueOf(aQty2);
                Validator.getQty(TheQty);
        }

        if (this.cboItem3.getSelectedIndex() != -1)
        {
                try
                {
                        txtQty3.commitEdit();
                }
                catch (java.text.ParseException e)
                {
                        Validator.setError("Invalid Quantity\n");
                }
                aQty3 = (Integer) txtQty3.getValue();
                String TheQty = String.valueOf(aQty3);
                Validator.getQty(TheQty);
        }

        if (this.cboItem1.getSelectedIndex() != -1)
        {
                Validator.getPrice(txtPrice1.getText());
        }

        if (this.cboItem2.getSelectedIndex() != -1)
        {
                Validator.getPrice(txtPrice2.getText());
        }

        if (this.cboItem3.getSelectedIndex() != -1)
        {
                Validator.getPrice(txtPrice3.getText());
        }

        // do the same thing for the other 2 comboboxes

        // if there are errors then display them to the user
        // otherwise start creating the objects for the order
        if (Validator.getError().length() != 0)
        {
                JOptionPane.showMessageDialog(null, "An Error Occured:\n\n" + Validato…

        }
        else
        {
                Customer customer = new Customer(firstName, lastName, street, city, st…

                Order order = new Order(orderID);
                order.setOrderCustomer(customer);
                NumberFormat nf = NumberFormat.getCurrencyInstance();

                if (this.cboItem1.getSelectedIndex() != -1)
                {
                        Product product1 = menu.getMenu()[this.cboItem1.getSelectedInd…
                        order.setOrderProduct(product1, aQty1);
                        String formatedLineTotal = nf.format(order.getLineTotal(0));
                        this.txtLineTotal1.setText(formatedLineTotal);
                }
                if (this.cboItem2.getSelectedIndex() != -1)
                {
                        Product product2 = menu.getMenu()[this.cboItem2.getSelectedInd…
                        order.setOrderProduct(product2, aQty2);
                        if (this.cboItem1.getSelectedIndex() != -1)
                        {
```

```
                                        String formatedLineTotal = nf.format(order.getLineTota…
                                        this.txtLineTotal2.setText(formatedLineTotal);
                                }
                                else
                                {
                                        String formatedLineTotal = nf.format(order.getLineTota…
                                        this.txtLineTotal2.setText(formatedLineTotal);
                                }
                        }
                        if (this.cboItem3.getSelectedIndex() != -1)
                        {
                                Product product3 = menu.getMenu()[this.cboItem3.getSelectedInd…
                                order.setOrderProduct(product3, aQty3);
                                if (this.cboItem1.getSelectedIndex() != -1 && this.cboItem2.ge…
                                {
                                        String formatedLineTotal = nf.format(order.getLineTota…
                                        this.txtLineTotal3.setText(formatedLineTotal);
                                }
                                else
                                {
                                        if (this.cboItem2.getSelectedIndex() != -1 || this.cbo…
                                        {
                                                String formatedLineTotal = nf.format(order.get…
                                                this.txtLineTotal3.setText(formatedLineTotal);
                                        }
                                        else
                                        {
                                                if (this.cboItem2.getSelectedIndex() == -1 && …
                                                {
                                                        String formatedLineTotal = nf.format(o…
                                                        this.txtLineTotal3.setText(formatedLin…
                                                }
                                        }
                                }
                        }

                        jz.addOrder(order);
                        jz.setClerk(orderID);
                        jz.processOrder(orderID);

                        String formatedSubTotal = nf.format(order.getSubTotal());
                        this.txtSubtotal.setText(formatedSubTotal);

                        String formatedTax = nf.format(order.getTax());
                        this.txtTax.setText(formatedTax);

                        String formatedTotal = nf.format(order.getTotal());
                        this.txtTotal.setText(formatedTotal);
                        // TODO
                        // INSTEAD OF GET RECEIPT YOU NEED TO GREATE METHODS THAT
                        // WILL RETURN INDIVIDUAL VALUES
                        // FOR EACH LINE ITEM RETRIEVE THE TOTALS
                        // FOR THE WHOLE ORDER RETRIEVE THE SUBTOTAL, TAX, TOTAL
                        // DISPLAY THESE RESULTS IN THE APPROPRIATE BOXES INSTEAD OF THE
                        // JOPTIONPANE BOX

                        // JOptionPane.showMessageDialog(null, jz.getReceipt(orderID));
                        JOptionPane.showMessageDialog(null, "Order " + orderID + " was Saved");
                }

        }

        private void deleteOrder()
```

```
    {
            String orderID = JOptionPane.showInputDialog("Enter Order ID to Delete");
            String result = jz.deleteOrder(orderID);

            JOptionPane.showMessageDialog(null, result);
    }

    private void findOrder()
    {
            // find a particular order

            // TODO
            // Fill the screen with the order that was retrieved
            // All the fields should be blanked out and filled with the values from
            // the new order

            String input = txtOrderID.getText();
            Order foundOrder = jz.findOrder(input);
            if (foundOrder != null)
            {
                    Customer foundCustomer = foundOrder.getCustomer();

                    txtFirstName.setText(foundCustomer.getFirstName());
                    txtLastName.setText(foundCustomer.getLastName());
                    txtStreet.setText(foundCustomer.getAddress());
                    txtCity.setText(foundCustomer.getCity());
                    txtState.setText(foundCustomer.getState());
                    txtZip.setText(foundCustomer.getZip());
                    txtPhone.setText(foundCustomer.getPhone());
                    txtMemberShip.setText(foundCustomer.getMemberShip());
                    try
                    {
                            cboItem1.setSelectedItem(foundOrder.getOrderProduct(0));
                            cboItem2.setSelectedItem(foundOrder.getOrderProduct(1));
                            cboItem3.setSelectedItem(foundOrder.getOrderProduct(2));
                    }
                    catch (Exception e)
                    {
                    }
                    try
                    {
                            txtQty1.setValue(foundOrder.getOrderQuantity(0));
                            txtQty2.setValue(foundOrder.getOrderQuantity(1));
                            txtQty3.setValue(foundOrder.getOrderQuantity(2));
                    }
                    catch (Exception e)
                    {
                    }
                    try
                    {
                            NumberFormat nf = NumberFormat.getCurrencyInstance();
                            txtLineTotal1.setText(nf.format((foundOrder.getLineTotal(0))));
                            txtLineTotal2.setText(nf.format((foundOrder.getLineTotal(1))));
                            txtLineTotal3.setText(nf.format((foundOrder.getLineTotal(2))));
                    }
                    catch (Exception e)
                    {

                    }
                    NumberFormat nf = NumberFormat.getCurrencyInstance();
                    txtSubtotal.setText(nf.format((foundOrder.getSubTotal())));
                    txtTax.setText(nf.format((foundOrder.getTax())));
                    txtTotal.setText(nf.format((foundOrder.getTotal())));
```

```
        }
        else
        {

                JOptionPane.showMessageDialog(null, "Order Not Found");

        }
}

private void clearOrders()
{
        // clear orders
        jz.clearOrders();
        JOptionPane.showMessageDialog(null, "Orders Cleared!");
}

private void printOrders()
{
        // print summary of orders
        String result = jz.getOrdersSummary();
        if (result.length() == 0)
        {
                JOptionPane.showMessageDialog(null, "No Orders Available!");
        }
        else
        {
                JOptionPane.showMessageDialog(null, jz.getOrdersSummary());
        }
}

public void resetForm()
{
        // TODO
        // ADD code here to reset the form to a blank state

        txtOrderID.setText("");
        txtFirstName.setText("");
        txtLastName.setText("");
        txtMemberShip.setText("");
        txtPhone.setText("");
        txtStreet.setText("");
        txtState.setText("");
        txtCity.setText("");
        txtZip.setText("");
        txtQty1.setValue(0);
        txtQty2.setValue(0);
        txtQty3.setValue(0);
        txtPrice1.setText("");
        txtPrice2.setText("");
        txtPrice3.setText("");
        txtLineTotal1.setText("");
        txtLineTotal2.setText("");
        txtLineTotal3.setText("");
        txtSubtotal.setText("");
        txtTax.setText("");
        txtTotal.setText("");

        try
        {

                this.cboItem1.setSelectedIndex(-1);
                this.cboItem2.setSelectedIndex(-1);
                this.cboItem3.setSelectedIndex(-1);
                }
                catch (Exception ex)
                {
                        // ignore error
                }
        }

}
```