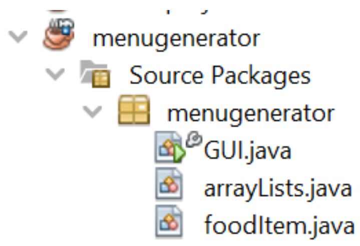
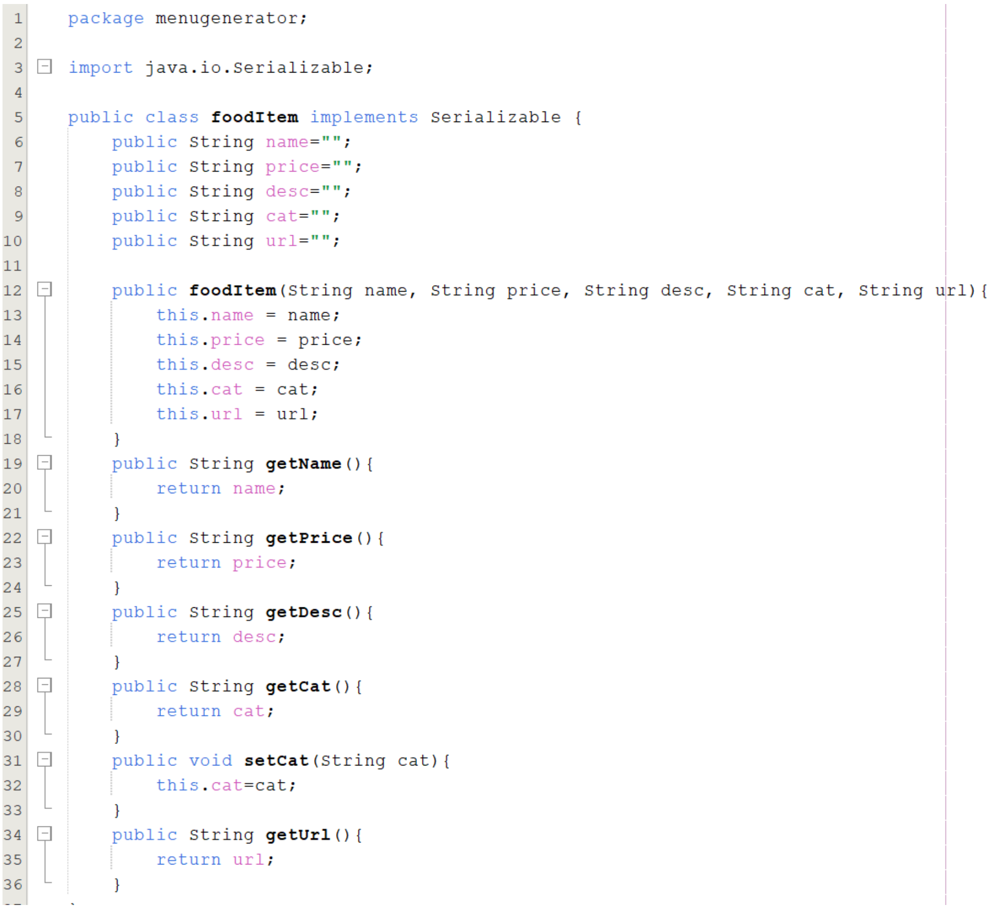
Appendix - III



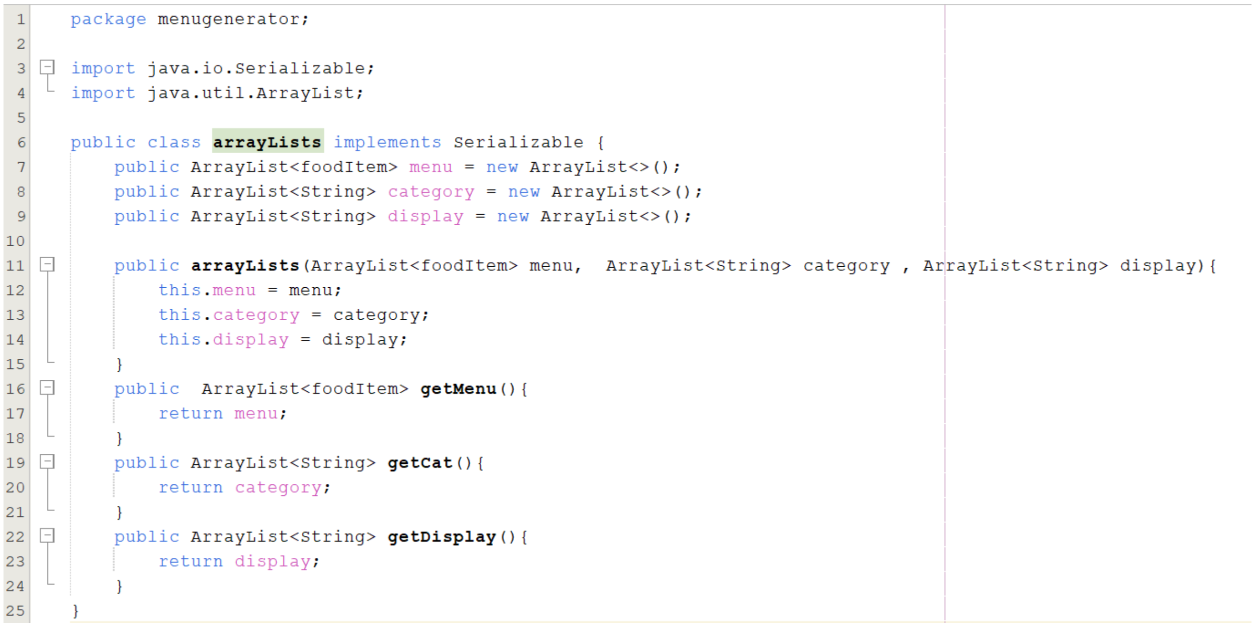
menugenerator
Source Packages
menugenerator
GUI.java
arrayLists.java
foodItem.java

foodItem.java



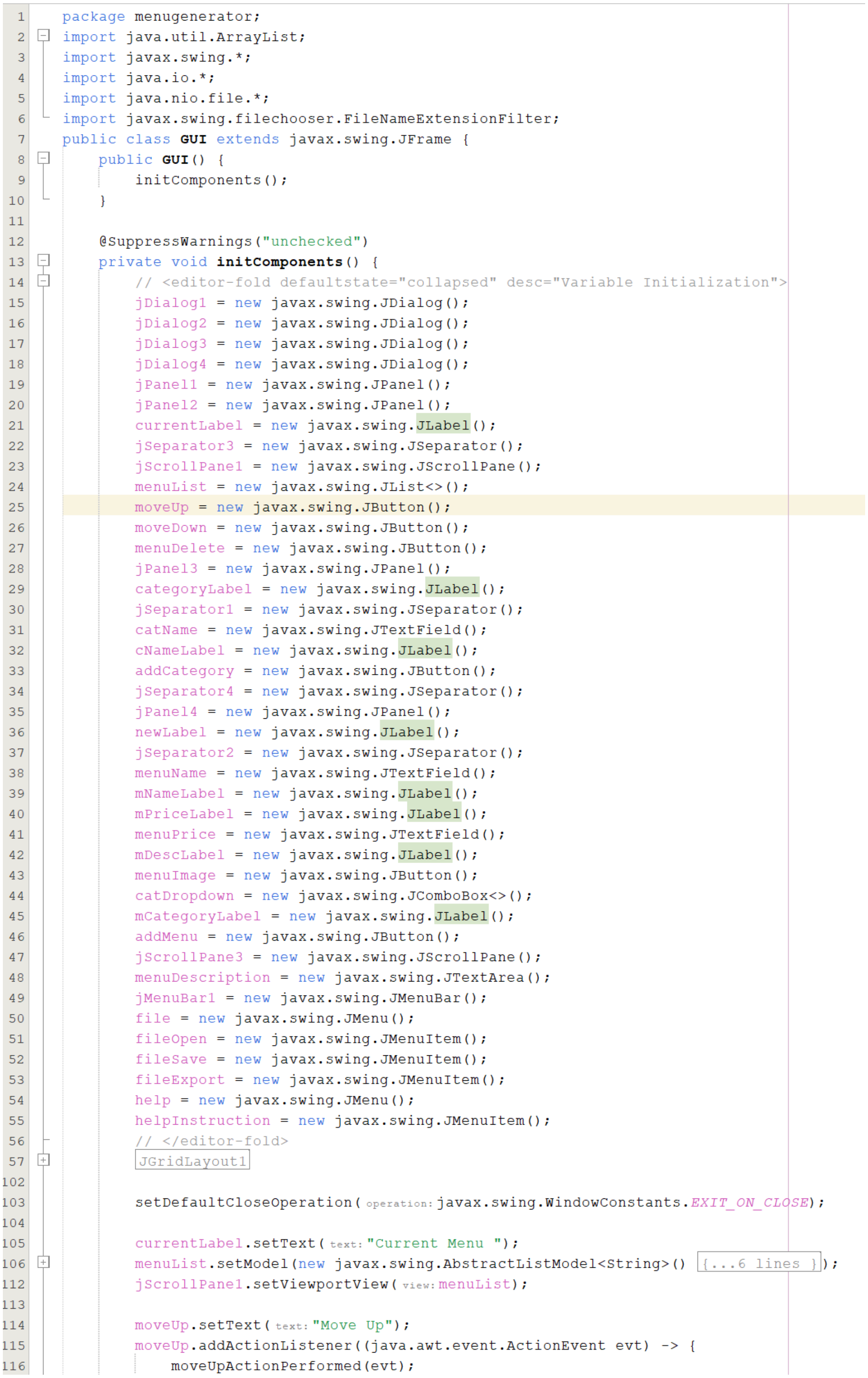
```java
package menugenerator;

import java.io.Serializable;

public class foodItem implements Serializable {
    public String name="";
    public String price="";
    public String desc="";
    public String cat="";
    public String url="";

    public foodItem(String name, String price, String desc, String cat, String url){
        this.name = name;
        this.price = price;
        this.desc = desc;
        this.cat = cat;
        this.url = url;
    }
    public String getName(){
        return name;
    }
    public String getPrice(){
        return price;
    }
    public String getDesc(){
        return desc;
    }
    public String getCat(){
        return cat;
    }
    public void setCat(String cat){
        this.cat=cat;
    }
    public String getUrl(){
        return url;
    }
```

arrayLists.java

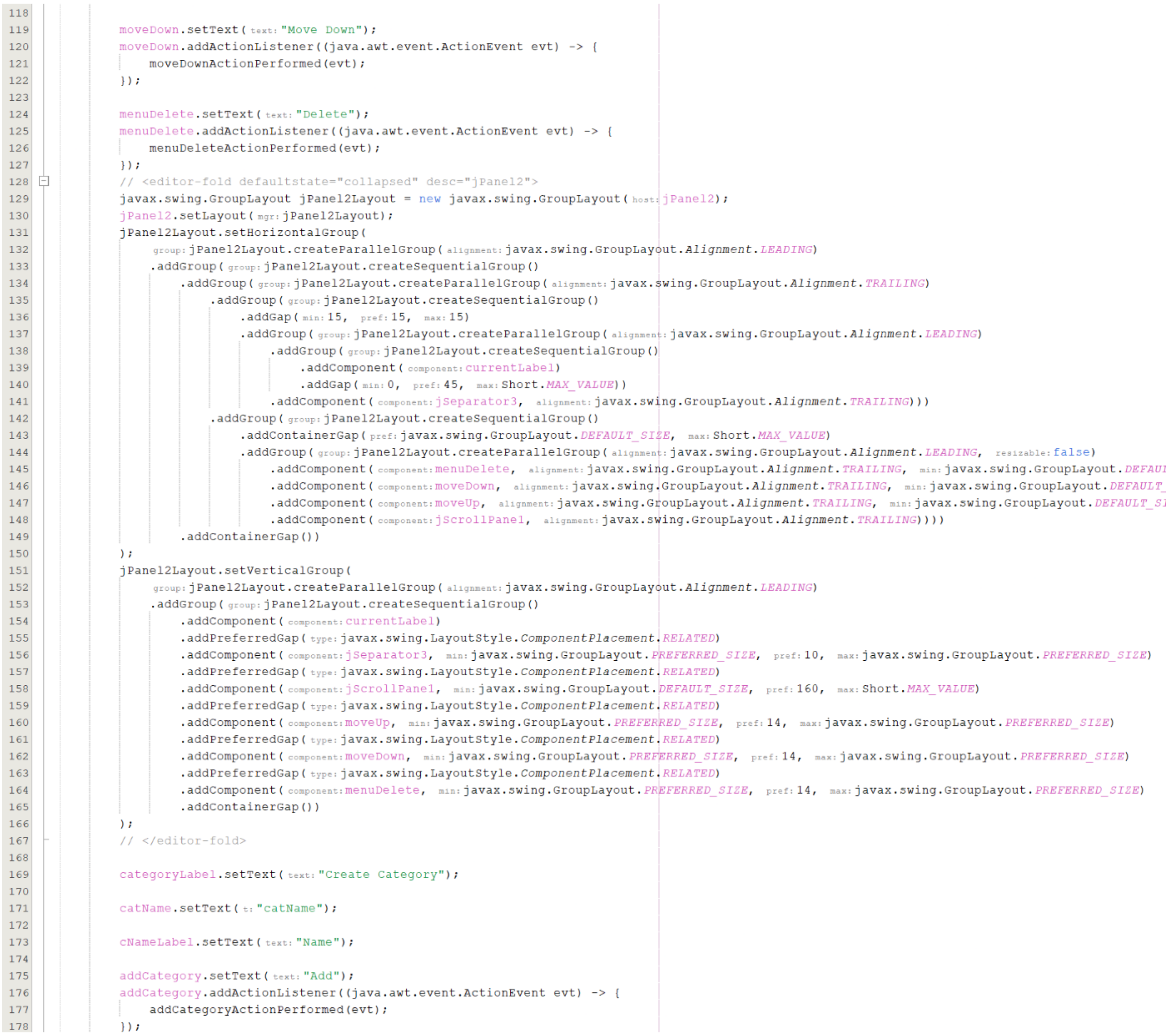


```java
package menugenerator;

import java.io.Serializable;
import java.util.ArrayList;

public class arrayLists implements Serializable {
    public ArrayList<foodItem> menu = new ArrayList<>();
    public ArrayList<String> category = new ArrayList<>();
    public ArrayList<String> display = new ArrayList<>();

    public arrayLists(ArrayList<foodItem> menu,  ArrayList<String> category , ArrayList<String> display){
        this.menu = menu;
        this.category = category;
        this.display = display;
    }
    public  ArrayList<foodItem> getMenu(){
        return menu;
    }
    public ArrayList<String> getCat(){
        return category;
    }
    public ArrayList<String> getDisplay(){
        return display;
    }
}
```

GUI.java

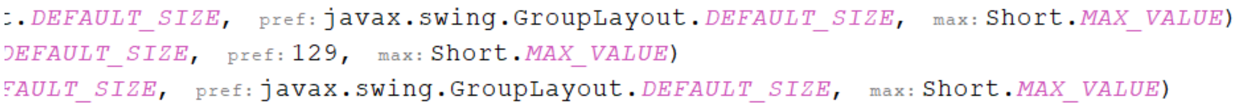


```java
package menugenerator;
import java.util.ArrayList;
import javax.swing.*;
import java.io.*;
import java.nio.file.*;
import javax.swing.filechooser.FileNameExtensionFilter;
public class GUI extends javax.swing.JFrame {
    public GUI() {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    private void initComponents() {
        // <editor-fold defaultstate="collapsed" desc="Variable Initialization">
        jDialog1 = new javax.swing.JDialog();
        jDialog2 = new javax.swing.JDialog();
        jDialog3 = new javax.swing.JDialog();
        jDialog4 = new javax.swing.JDialog();
        jPanel1 = new javax.swing.JPanel();
        jPanel2 = new javax.swing.JPanel();
        currentLabel = new javax.swing.JLabel();
        jSeparator3 = new javax.swing.JSeparator();
        jScrollPane1 = new javax.swing.JScrollPane();
        menuList = new javax.swing.JList<>();
        moveUp = new javax.swing.JButton();
        moveDown = new javax.swing.JButton();
        menuDelete = new javax.swing.JButton();
        jPanel3 = new javax.swing.JPanel();
        categoryLabel = new javax.swing.JLabel();
        jSeparator1 = new javax.swing.JSeparator();
        catName = new javax.swing.JTextField();
        cNameLabel = new javax.swing.JLabel();
        addCategory = new javax.swing.JButton();
        jSeparator4 = new javax.swing.JSeparator();
        jPanel4 = new javax.swing.JPanel();
        newLabel = new javax.swing.JLabel();
        jSeparator2 = new javax.swing.JSeparator();
        menuName = new javax.swing.JTextField();
        mNameLabel = new javax.swing.JLabel();
        mPriceLabel = new javax.swing.JLabel();
        menuPrice = new javax.swing.JTextField();
        mDescLabel = new javax.swing.JLabel();
        menuImage = new javax.swing.JButton();
        catDropdown = new javax.swing.JComboBox<>();
        mCategoryLabel = new javax.swing.JLabel();
        addMenu = new javax.swing.JButton();
        jScrollPane3 = new javax.swing.JScrollPane();
        menuDescription = new javax.swing.JTextArea();
        jMenuBar1 = new javax.swing.JMenuBar();
        file = new javax.swing.JMenu();
        fileOpen = new javax.swing.JMenuItem();
        fileSave = new javax.swing.JMenuItem();
        fileExport = new javax.swing.JMenuItem();
        help = new javax.swing.JMenu();
        helpInstruction = new javax.swing.JMenuItem();
        // </editor-fold>
        JGridLayout1

        setDefaultCloseOperation( operation: javax.swing.WindowConstants.EXIT_ON_CLOSE);

        currentLabel.setText( text: "Current Menu ");
        menuList.setModel(new javax.swing.AbstractListModel<String>() {...6 lines });
        jScrollPane1.setViewportView( view: menuList);

        moveUp.setText( text: "Move Up");
        moveUp.addActionListener((java.awt.event.ActionEvent evt) -> {
            moveUpActionPerformed(evt);
```

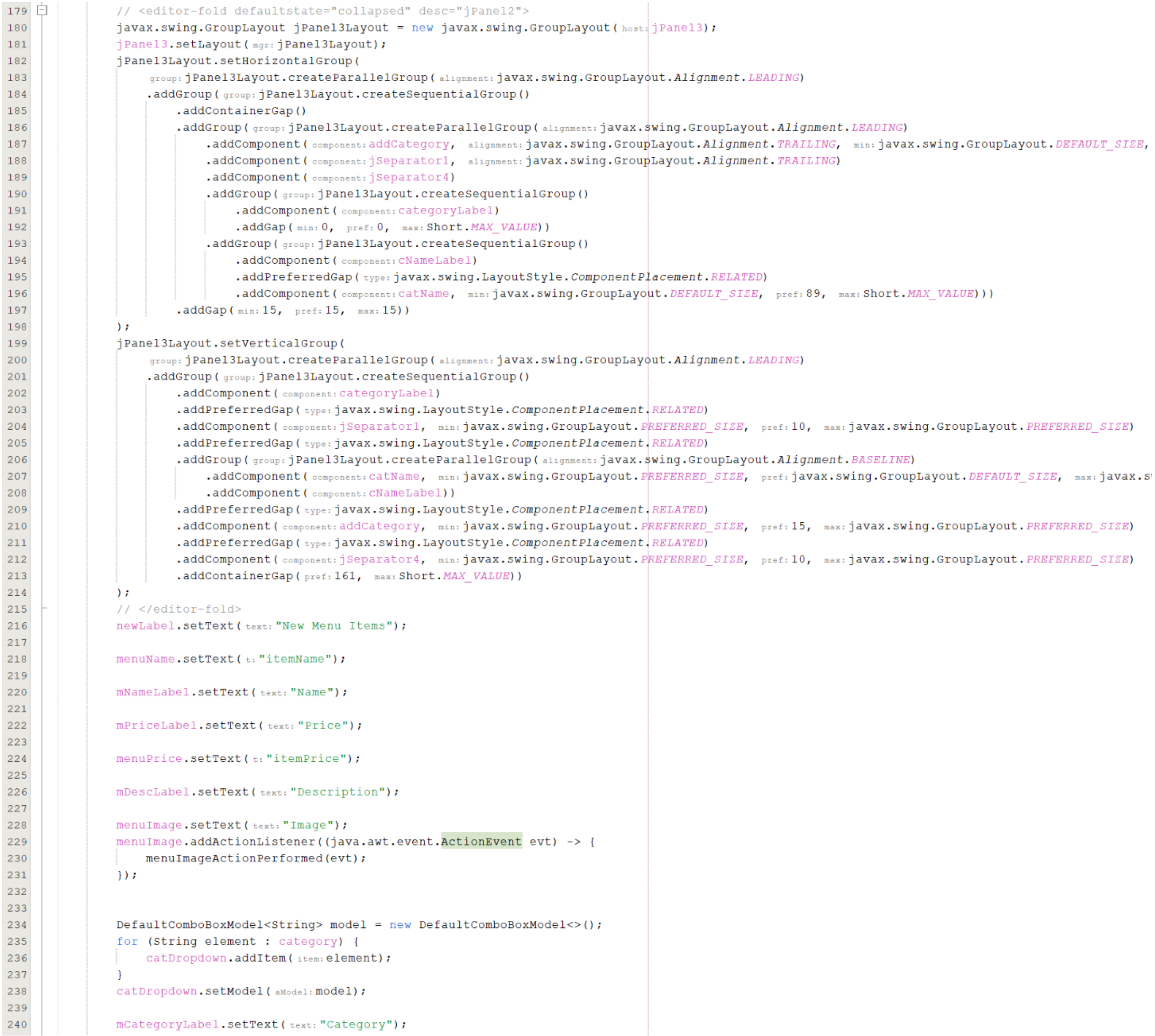


```
118
119     moveDown.setText( text: "Move Down");
120     moveDown.addActionListener((java.awt.event.ActionEvent evt) -> {
121         moveDownActionPerformed(evt);
122     });
123
124     menuDelete.setText( text: "Delete");
125     menuDelete.addActionListener((java.awt.event.ActionEvent evt) -> {
126         menuDeleteActionPerformed(evt);
127     });
128     // <editor-fold defaultstate="collapsed" desc="jPanel2">
129     javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout( host: jPanel2);
130     jPanel2.setLayout( mgr: jPanel2Layout);
131     jPanel2Layout.setHorizontalGroup(
132          group: jPanel2Layout.createParallelGroup( alignment: javax.swing.GroupLayout.Alignment.LEADING)
133         .addGroup( group: jPanel2Layout.createSequentialGroup()
134             .addGroup( group: jPanel2Layout.createParallelGroup( alignment: javax.swing.GroupLayout.Alignment.TRAILING)
135                 .addGroup( group: jPanel2Layout.createSequentialGroup()
136                     .addGap( min: 15,  pref: 15,  max: 15)
137                     .addGroup( group: jPanel2Layout.createParallelGroup( alignment: javax.swing.GroupLayout.Alignment.LEADING)
138                         .addGroup( group: jPanel2Layout.createSequentialGroup()
139                             .addComponent( component: currentLabel)
140                             .addGap( min: 0,  pref: 45,  max: Short.MAX_VALUE))
141                         .addComponent( component: jSeparator3,  alignment: javax.swing.GroupLayout.Alignment.TRAILING)))
142                 .addGroup( group: jPanel2Layout.createSequentialGroup()
143                     .addContainerGap( pref: javax.swing.GroupLayout.DEFAULT_SIZE,  max: Short.MAX_VALUE)
144                     .addGroup( group: jPanel2Layout.createParallelGroup( alignment: javax.swing.GroupLayout.Alignment.LEADING,  resizable: false)
145                         .addComponent( component: menuDelete,  alignment: javax.swing.GroupLayout.Alignment.TRAILING,  min: javax.swing.GroupLayout.DEFAU
146                         .addComponent( component: moveDown,  alignment: javax.swing.GroupLayout.Alignment.TRAILING,  min: javax.swing.GroupLayout.DEFAULT_
147                         .addComponent( component: moveUp,  alignment: javax.swing.GroupLayout.Alignment.TRAILING,  min: javax.swing.GroupLayout.DEFAULT_S
148                         .addComponent( component: jScrollPane1,  alignment: javax.swing.GroupLayout.Alignment.TRAILING))))
149             .addContainerGap())
150     );
151     jPanel2Layout.setVerticalGroup(
152          group: jPanel2Layout.createParallelGroup( alignment: javax.swing.GroupLayout.Alignment.LEADING)
153         .addGroup( group: jPanel2Layout.createSequentialGroup()
154             .addComponent( component: currentLabel)
155             .addPreferredGap( type: javax.swing.LayoutStyle.ComponentPlacement.RELATED)
156             .addComponent( component: jSeparator3,  min: javax.swing.GroupLayout.PREFERRED_SIZE,  pref: 10,  max: javax.swing.GroupLayout.PREFERRED_SIZE)
157             .addPreferredGap( type: javax.swing.LayoutStyle.ComponentPlacement.RELATED)
158             .addComponent( component: jScrollPane1,  min: javax.swing.GroupLayout.DEFAULT_SIZE,  pref: 160,  max: Short.MAX_VALUE)
159             .addPreferredGap( type: javax.swing.LayoutStyle.ComponentPlacement.RELATED)
160             .addComponent( component: moveUp,  min: javax.swing.GroupLayout.PREFERRED_SIZE,  pref: 14,  max: javax.swing.GroupLayout.PREFERRED_SIZE)
161             .addPreferredGap( type: javax.swing.LayoutStyle.ComponentPlacement.RELATED)
162             .addComponent( component: moveDown,  min: javax.swing.GroupLayout.PREFERRED_SIZE,  pref: 14,  max: javax.swing.GroupLayout.PREFERRED_SIZE)
163             .addPreferredGap( type: javax.swing.LayoutStyle.ComponentPlacement.RELATED)
164             .addComponent( component: menuDelete,  min: javax.swing.GroupLayout.PREFERRED_SIZE,  pref: 14,  max: javax.swing.GroupLayout.PREFERRED_SIZE)
165             .addContainerGap())
166     );
167     // </editor-fold>
168
169     categoryLabel.setText( text: "Create Category");
170
171     catName.setText( t: "catName");
172
173     cNameLabel.setText( text: "Name");
174
175     addCategory.setText( text: "Add");
176     addCategory.addActionListener((java.awt.event.ActionEvent evt) -> {
177         addCategoryActionPerformed(evt);
178     });
```

Lines 145-147 Continued:

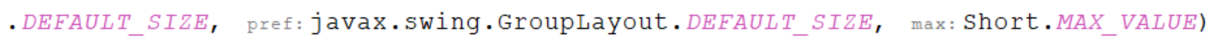


```
.DEFAULT_SIZE,  pref: javax.swing.GroupLayout.DEFAULT_SIZE,  max: Short.MAX_VALUE)
DEFAULT_SIZE,  pref: 129,  max: Short.MAX_VALUE)
FAULT_SIZE,  pref: javax.swing.GroupLayout.DEFAULT_SIZE,  max: Short.MAX_VALUE)
```



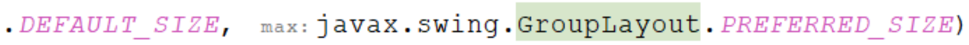
```
179 // <editor-fold defaultstate="collapsed" desc="jPanel2">
180 javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout( host: jPanel3);
181 jPanel3.setLayout( mgr: jPanel3Layout);
182 jPanel3Layout.setHorizontalGroup(
183     group: jPanel3Layout.createParallelGroup( alignment: javax.swing.GroupLayout.Alignment.LEADING)
184     .addGroup( group: jPanel3Layout.createSequentialGroup()
185         .addContainerGap()
186         .addGroup( group: jPanel3Layout.createParallelGroup( alignment: javax.swing.GroupLayout.Alignment.LEADING)
187             .addComponent( component: addCategory,  alignment: javax.swing.GroupLayout.Alignment.TRAILING,  min: javax.swing.GroupLayout.DEFAULT_SIZE,
188             .addComponent( component: jSeparator1,  alignment: javax.swing.GroupLayout.Alignment.TRAILING)
189             .addComponent( component: jSeparator4)
190             .addGroup( group: jPanel3Layout.createSequentialGroup()
191                 .addComponent( component: categoryLabel)
192                 .addGap( min: 0,  pref: 0,  max: Short.MAX_VALUE))
193             .addGroup( group: jPanel3Layout.createSequentialGroup()
194                 .addComponent( component: cNameLabel)
195                 .addPreferredGap( type: javax.swing.LayoutStyle.ComponentPlacement.RELATED)
196                 .addComponent( component: catName,  min: javax.swing.GroupLayout.DEFAULT_SIZE,  pref: 89,  max: Short.MAX_VALUE)))
197         .addGap( min: 15,  pref: 15,  max: 15))
198 );
199 jPanel3Layout.setVerticalGroup(
200     group: jPanel3Layout.createParallelGroup( alignment: javax.swing.GroupLayout.Alignment.LEADING)
201     .addGroup( group: jPanel3Layout.createSequentialGroup()
202         .addComponent( component: categoryLabel)
203         .addPreferredGap( type: javax.swing.LayoutStyle.ComponentPlacement.RELATED)
204         .addComponent( component: jSeparator1,  min: javax.swing.GroupLayout.PREFERRED_SIZE,  pref: 10,  max: javax.swing.GroupLayout.PREFERRED_SIZE)
205         .addPreferredGap( type: javax.swing.LayoutStyle.ComponentPlacement.RELATED)
206         .addGroup( group: jPanel3Layout.createParallelGroup( alignment: javax.swing.GroupLayout.Alignment.BASELINE)
207             .addComponent( component: catName,  min: javax.swing.GroupLayout.PREFERRED_SIZE,  pref: javax.swing.GroupLayout.DEFAULT_SIZE,  max: javax.s
208             .addComponent( component: cNameLabel))
209         .addPreferredGap( type: javax.swing.LayoutStyle.ComponentPlacement.RELATED)
210         .addComponent( component: addCategory,  min: javax.swing.GroupLayout.PREFERRED_SIZE,  pref: 15,  max: javax.swing.GroupLayout.PREFERRED_SIZE)
211         .addPreferredGap( type: javax.swing.LayoutStyle.ComponentPlacement.RELATED)
212         .addComponent( component: jSeparator4,  min: javax.swing.GroupLayout.PREFERRED_SIZE,  pref: 10,  max: javax.swing.GroupLayout.PREFERRED_SIZE)
213         .addContainerGap( pref: 161,  max: Short.MAX_VALUE))
214 );
215 // </editor-fold>
216 newLabel.setText( text: "New Menu Items");
217
218 menuName.setText( t: "itemName");
219
220 mNameLabel.setText( text: "Name");
221
222 mPriceLabel.setText( text: "Price");
223
224 menuPrice.setText( t: "itemPrice");
225
226 mDescLabel.setText( text: "Description");
227
228 menuImage.setText( text: "Image");
229 menuImage.addActionListener((java.awt.event.ActionEvent evt) -> {
230     menuImageActionPerformed(evt);
231 });
232
233
234 DefaultComboBoxModel<String> model = new DefaultComboBoxModel<>();
235 for (String element : category) {
236     catDropdown.addItem( item: element);
237 }
238 catDropdown.setModel( aModel: model);
239
240 mCategoryLabel.setText( text: "Category");
```

Line 187 Continued:

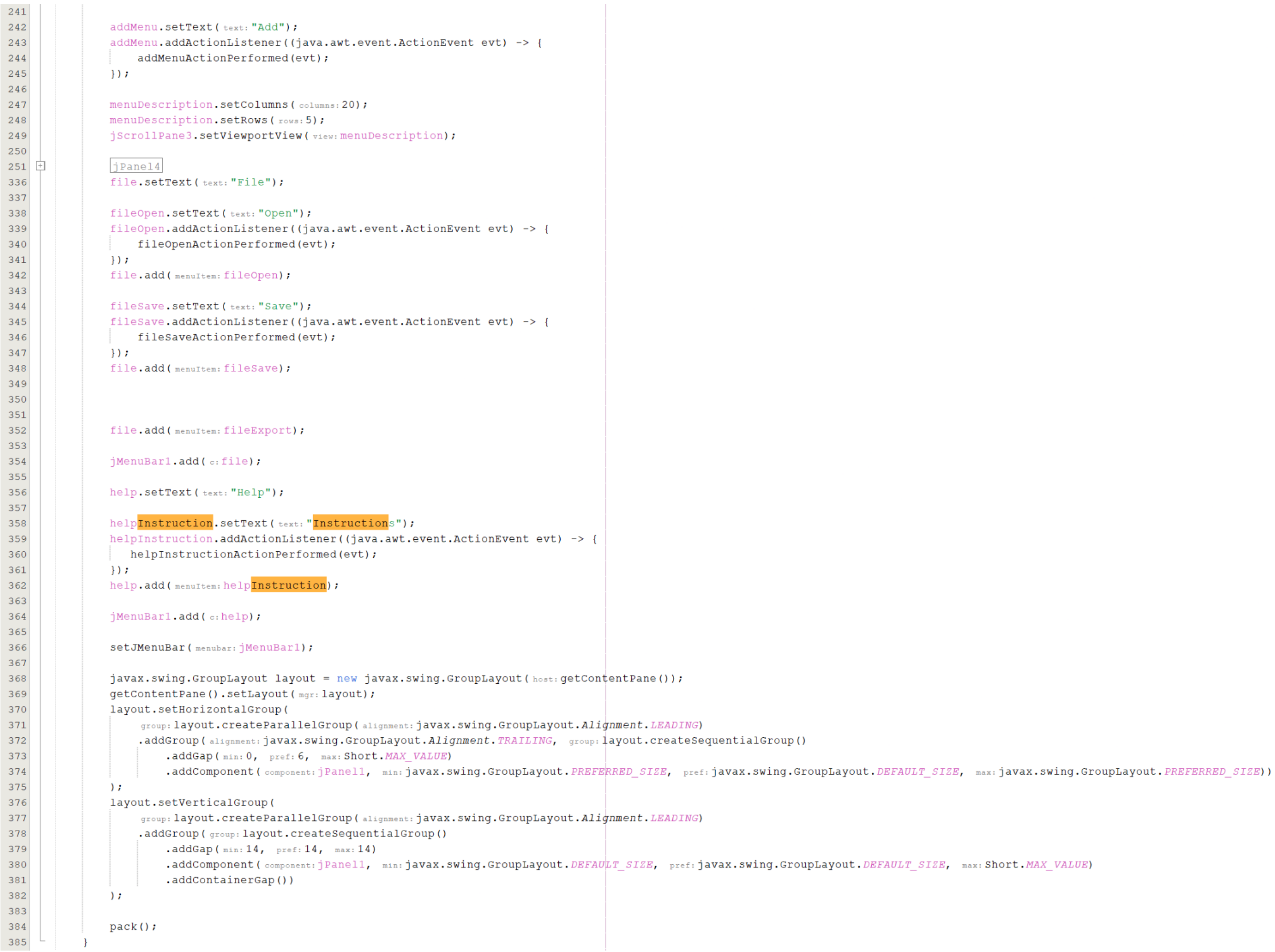


```
.DEFAULT_SIZE,  pref: javax.swing.GroupLayout.DEFAULT_SIZE,  max: Short.MAX_VALUE)
```

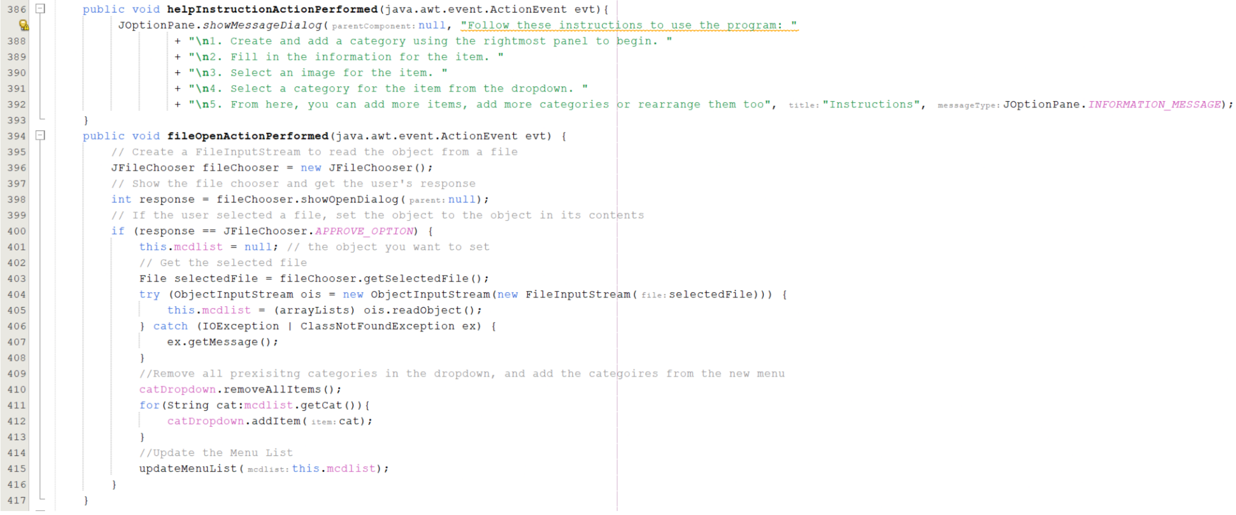
Line 207 Continued:



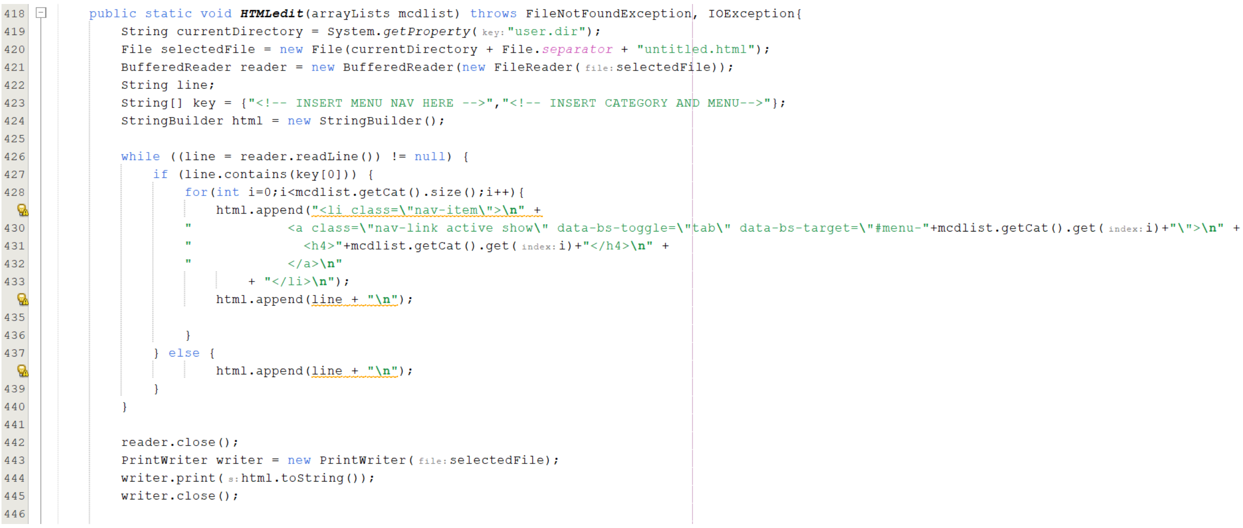
```
.DEFAULT_SIZE,  max: javax.swing.GroupLayout.PREFERRED_SIZE)
```



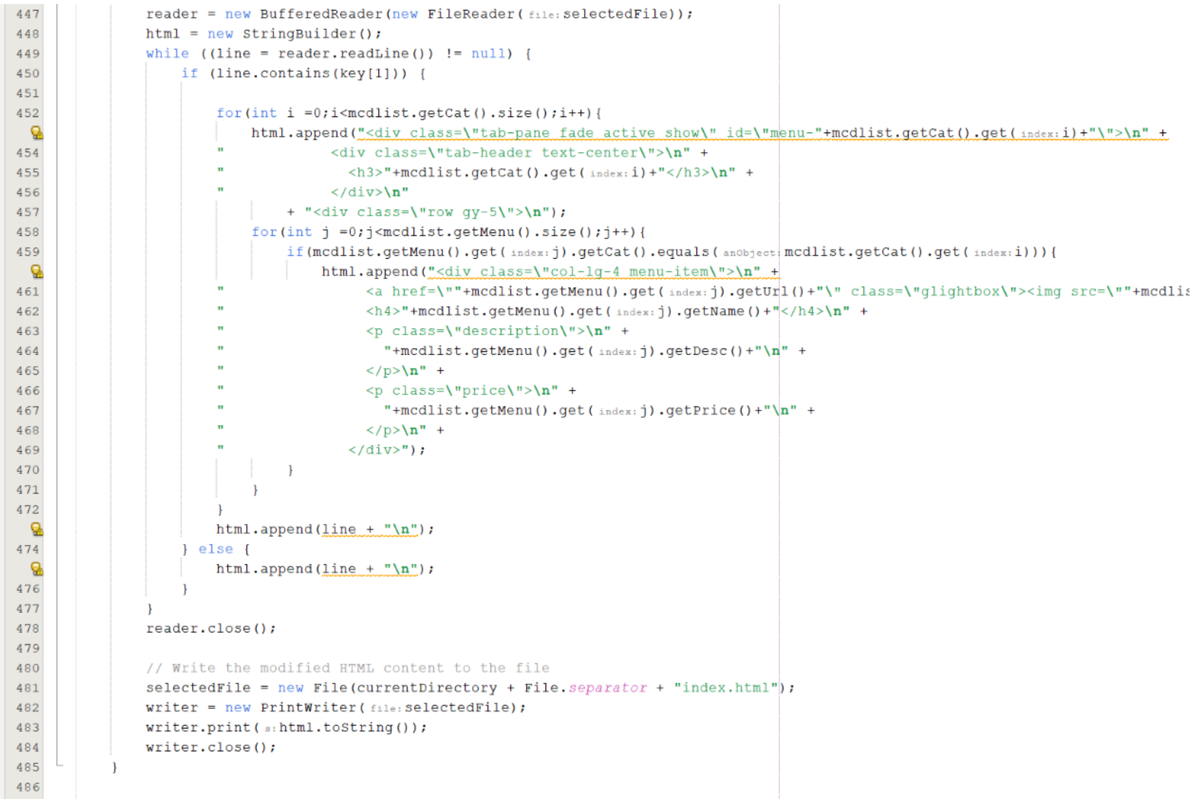
```
    addMenu.setText( text: "Add");
    addMenu.addActionListener((java.awt.event.ActionEvent evt) -> {
        addMenuActionPerformed(evt);
    });

    menuDescription.setColumns( columns: 20);
    menuDescription.setRows( rows: 5);
    jScrollPane3.setViewportView( view: menuDescription);

    jPanel4
    file.setText( text: "File");

    fileOpen.setText( text: "Open");
    fileOpen.addActionListener((java.awt.event.ActionEvent evt) -> {
        fileOpenActionPerformed(evt);
    });
    file.add( menuItem: fileOpen);

    fileSave.setText( text: "Save");
    fileSave.addActionListener((java.awt.event.ActionEvent evt) -> {
        fileSaveActionPerformed(evt);
    });
    file.add( menuItem: fileSave);



    file.add( menuItem: fileExport);

    jMenuBar1.add( c: file);

    help.setText( text: "Help");

    helpInstruction.setText( text: "Instructions");
    helpInstruction.addActionListener((java.awt.event.ActionEvent evt) -> {
       helpInstructionActionPerformed(evt);
    });
    help.add( menuItem: helpInstruction);

    jMenuBar1.add( c: help);

    setJMenuBar( menubar: jMenuBar1);

    javax.swing.GroupLayout layout = new javax.swing.GroupLayout( host: getContentPane());
    getContentPane().setLayout( mgr: layout);
    layout.setHorizontalGroup(
         group: layout.createParallelGroup( alignment: javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup( alignment: javax.swing.GroupLayout.Alignment.TRAILING,  group: layout.createSequentialGroup()
            .addGap( min: 0,  pref: 6,  max: Short.MAX_VALUE)
            .addComponent( component: jPanel1,  min: javax.swing.GroupLayout.PREFERRED_SIZE,  pref: javax.swing.GroupLayout.DEFAULT_SIZE,  max: javax.swing.GroupLayout.PREFERRED_SIZE))
    );
    layout.setVerticalGroup(
         group: layout.createParallelGroup( alignment: javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup( group: layout.createSequentialGroup()
            .addGap( min: 14,  pref: 14,  max: 14)
            .addComponent( component: jPanel1,  min: javax.swing.GroupLayout.DEFAULT_SIZE,  pref: javax.swing.GroupLayout.DEFAULT_SIZE,  max: Short.MAX_VALUE)
            .addContainerGap())
    );

    pack();
}
```



```
public void helpInstructionActionPerformed(java.awt.event.ActionEvent evt){
     JOptionPane.showMessageDialog( parentComponent: null, "Follow these instructions to use the program: "
             + "\n1. Create and add a category using the rightmost panel to begin. "
             + "\n2. Fill in the information for the item. "
             + "\n3. Select an image for the item. "
             + "\n4. Select a category for the item from the dropdown. "
             + "\n5. From here, you can add more items, add more categories or rearrange them too",  title: "Instructions",  messageType: JOptionPane.INFORMATION_MESSAGE);
}
public void fileOpenActionPerformed(java.awt.event.ActionEvent evt) {
    // Create a FileInputStream to read the object from a file
    JFileChooser fileChooser = new JFileChooser();
    // Show the file chooser and get the user's response
    int response = fileChooser.showOpenDialog( parent: null);
    // If the user selected a file, set the object to the object in its contents
    if (response == JFileChooser.APPROVE_OPTION) {
        this.mcdlist = null; // the object you want to set
        // Get the selected file
        File selectedFile = fileChooser.getSelectedFile();
        try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream( file: selectedFile))) {
            this.mcdlist = (arrayLists) ois.readObject();
        } catch (IOException | ClassNotFoundException ex) {
            ex.getMessage();
        }
        //Remove all prexisitng categories in the dropdown, and add the categoires from the new menu
        catDropdown.removeAllItems();
        for(String cat:mcdlist.getCat()){
            catDropdown.addItem( item: cat);
        }
        //Update the Menu List
        updateMenuList( mcdlist: this.mcdlist);
    }
}
```



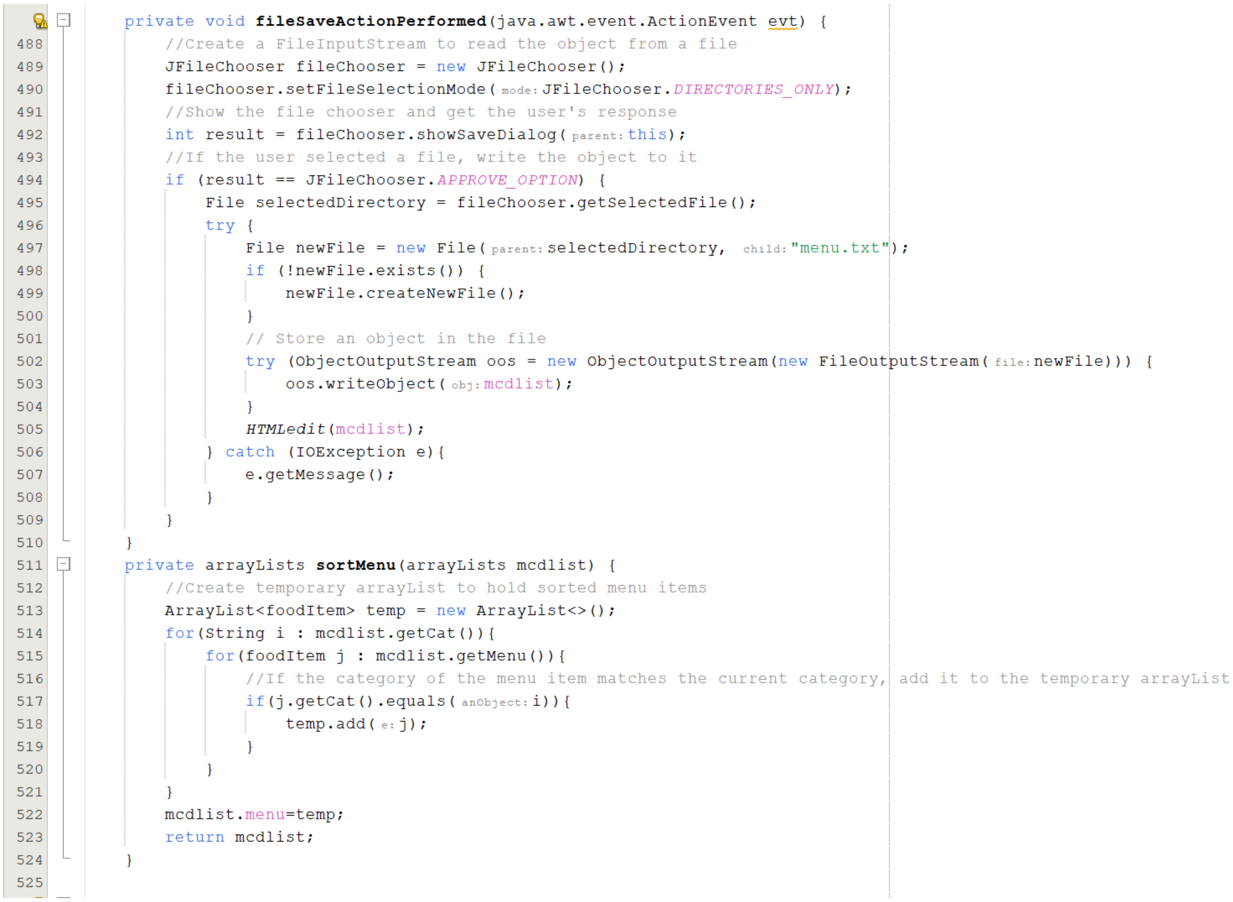
```
public static void HTMLedit(arrayLists mcdlist) throws FileNotFoundException, IOException{
    String currentDirectory = System.getProperty( key: "user.dir");
    File selectedFile = new File(currentDirectory + File.separator + "untitled.html");
    BufferedReader reader = new BufferedReader(new FileReader( file: selectedFile));
    String line;
    String[] key = {"<!-- INSERT MENU NAV HERE -->","<!-- INSERT CATEGORY AND MENU-->"};
    StringBuilder html = new StringBuilder();

    while ((line = reader.readLine()) != null) {
        if (line.contains(key[0])) {
            for(int i=0;i<mcdlist.getCat().size();i++){
                html.append("<li class=\"nav-item\">\n" +
            "            <a class=\"nav-link active show\" data-bs-toggle=\"tab\" data-bs-target=\"#menu-"+mcdlist.getCat().get( index: i)+"\">\n" +
            "              <h4>"+mcdlist.getCat().get( index: i)+"</h4>\n" +
            "            </a>\n"
                    + "</li>\n");
                html.append(line + "\n");

            }
        } else {
                html.append(line + "\n");
        }
    }

    reader.close();
    PrintWriter writer = new PrintWriter( file: selectedFile);
    writer.print( s: html.toString());
    writer.close();

```



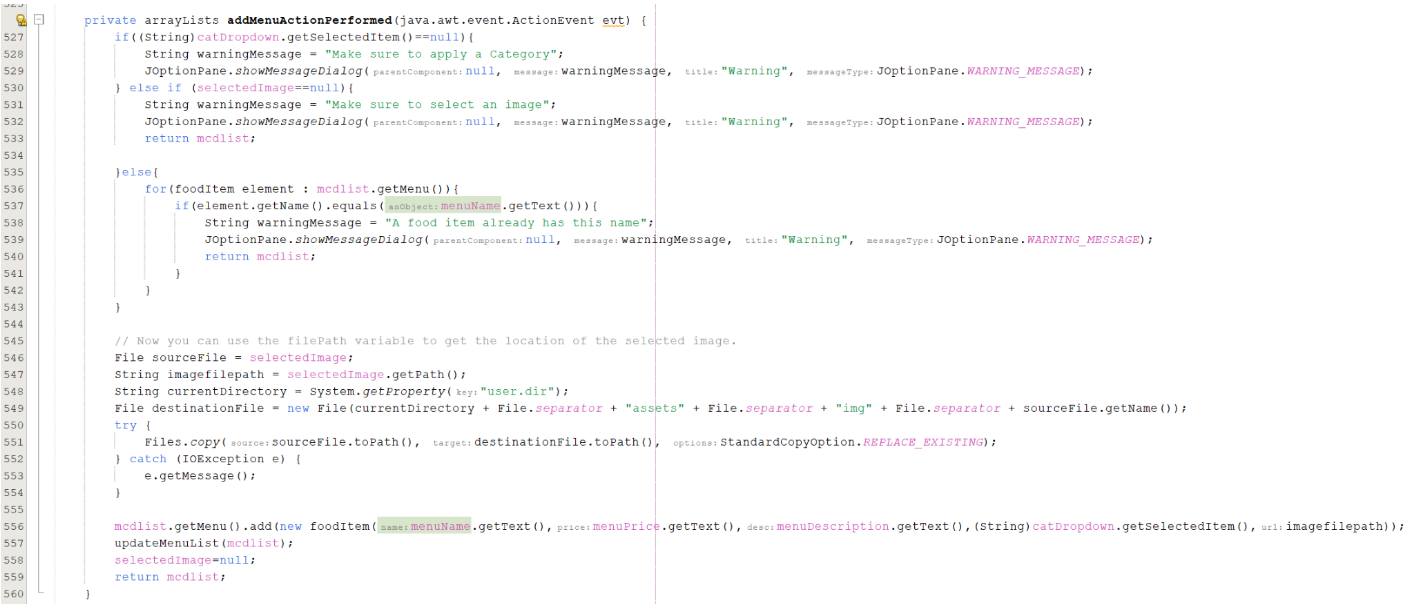
```
447             reader = new BufferedReader(new FileReader( file:selectedFile));
448             html = new StringBuilder();
449             while ((line = reader.readLine()) != null) {
450                 if (line.contains(key[1])) {
451 
452                     for(int i =0;i<mcdlist.getCat().size();i++){
                            html.append("<div class=\"tab-pane fade active show\" id=\"menu-"+mcdlist.getCat().get( index: i)+"\">\n" +
454                     "            <div class=\"tab-header text-center\">\n" +
455                     "              <h3>"+mcdlist.getCat().get( index: i)+"</h3>\n" +
456                     "            </div>\n"
457                             + "<div class=\"row gy-5\">\n");
458                         for(int j =0;j<mcdlist.getMenu().size();j++){
459                             if(mcdlist.getMenu().get( index: j).getCat().equals( anObject: mcdlist.getCat().get( index: i))){
                                    html.append("<div class=\"col-lg-4 menu-item\">\n" +
461                     "                <a href=\""+mcdlist.getMenu().get( index: j).getUrl()+"\" class=\"glightbox\"><img src=\""+mcdlis
462                     "                <h4>"+mcdlist.getMenu().get( index: j).getName()+"</h4>\n" +
463                     "                <p class=\"description\">\n" +
464                     "                  "+mcdlist.getMenu().get( index: j).getDesc()+"\n" +
465                     "                </p>\n" +
466                     "                <p class=\"price\">\n" +
467                     "                  "+mcdlist.getMenu().get( index: j).getPrice()+"\n" +
468                     "                </p>\n" +
469                     "              </div>");
470                             }
471                         }
472                     }
                        html.append(line + "\n");
474                 } else {
                        html.append(line + "\n");
476                 }
477             }
478             reader.close();
479 
480             // Write the modified HTML content to the file
481             selectedFile = new File(currentDirectory + File.separator + "index.html");
482             writer = new PrintWriter( file: selectedFile);
483             writer.print( s: html.toString());
484             writer.close();
485         }
486 
```

Line 461 Continued:

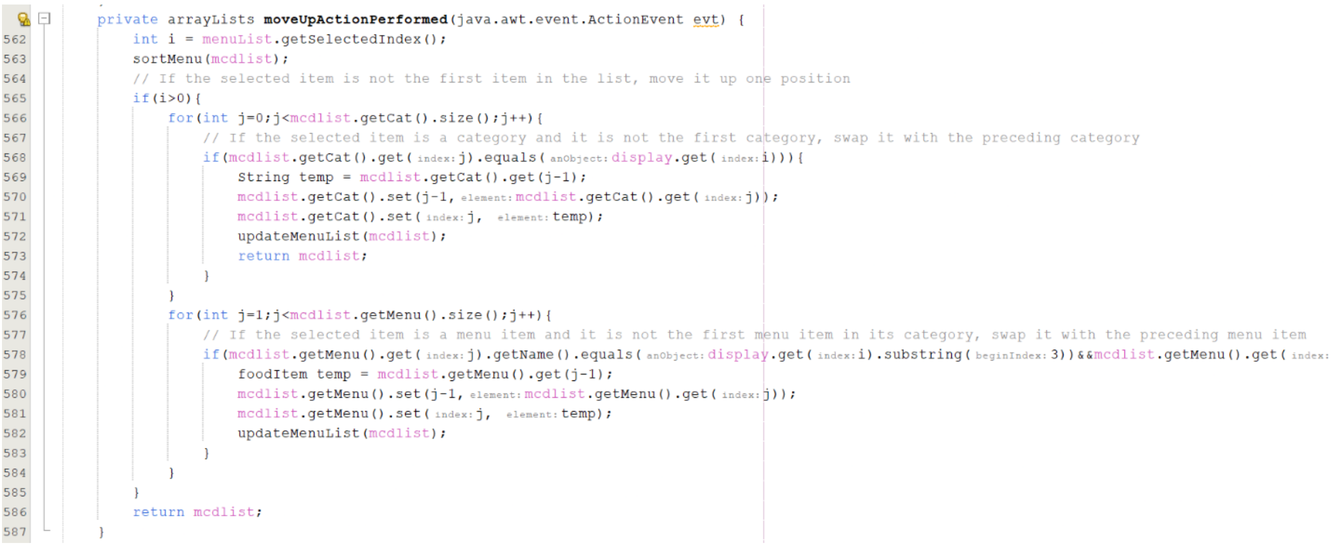
```
<img src=\""+mcdlist.getMenu().get( index: j).getUrl()+"\" class=\"menu-img img-fluid\" alt=\"\"></a>\n" +
```



```java
    private void fileSaveActionPerformed(java.awt.event.ActionEvent evt) {
        //Create a FileInputStream to read the object from a file
        JFileChooser fileChooser = new JFileChooser();
        fileChooser.setFileSelectionMode( mode: JFileChooser.DIRECTORIES_ONLY);
        //Show the file chooser and get the user's response
        int result = fileChooser.showSaveDialog( parent: this);
        //If the user selected a file, write the object to it
        if (result == JFileChooser.APPROVE_OPTION) {
            File selectedDirectory = fileChooser.getSelectedFile();
            try {
                File newFile = new File( parent: selectedDirectory,  child: "menu.txt");
                if (!newFile.exists()) {
                    newFile.createNewFile();
                }
                // Store an object in the file
                try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream( file: newFile))) {
                    oos.writeObject( obj: mcdlist);
                }
                HTMLedit(mcdlist);
            } catch (IOException e){
                e.getMessage();
            }
        }
    }
    private arrayLists sortMenu(arrayLists mcdlist) {
        //Create temporary arrayList to hold sorted menu items
        ArrayList<foodItem> temp = new ArrayList<>();
        for(String i : mcdlist.getCat()){
            for(foodItem j : mcdlist.getMenu()){
                //If the category of the menu item matches the current category, add it to the temporary arrayList
                if(j.getCat().equals( anObject: i)){
                    temp.add( e: j);
                }
            }
        }
        mcdlist.menu=temp;
        return mcdlist;
    }
```

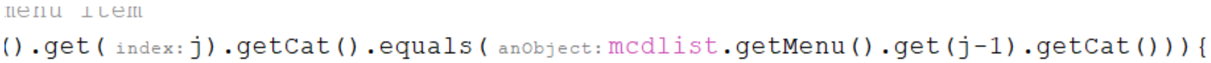


```java
    private arrayLists addMenuActionPerformed(java.awt.event.ActionEvent evt) {
        if((String)catDropdown.getSelectedItem()==null){
            String warningMessage = "Make sure to apply a Category";
            JOptionPane.showMessageDialog( parentComponent: null,  message: warningMessage,  title: "Warning",  messageType: JOptionPane.WARNING_MESSAGE);
        } else if (selectedImage==null){
            String warningMessage = "Make sure to select an image";
            JOptionPane.showMessageDialog( parentComponent: null,  message: warningMessage,  title: "Warning",  messageType: JOptionPane.WARNING_MESSAGE);
            return mcdlist;

        }else{
            for(foodItem element : mcdlist.getMenu()){
                if(element.getName().equals( anObject: menuName.getText())){
                    String warningMessage = "A food item already has this name";
                    JOptionPane.showMessageDialog( parentComponent: null,  message: warningMessage,  title: "Warning",  messageType: JOptionPane.WARNING_MESSAGE);
                    return mcdlist;
                }
            }
        }

        // Now you can use the filePath variable to get the location of the selected image.
        File sourceFile = selectedImage;
        String imagefilepath = selectedImage.getPath();
        String currentDirectory = System.getProperty( key: "user.dir");
        File destinationFile = new File(currentDirectory + File.separator + "assets" + File.separator + "img" + File.separator + sourceFile.getName());
        try {
            Files.copy( source: sourceFile.toPath(),  target: destinationFile.toPath(),  options: StandardCopyOption.REPLACE_EXISTING);
        } catch (IOException e) {
            e.getMessage();
        }

        mcdlist.getMenu().add(new foodItem( name: menuName.getText(), price: menuPrice.getText(), desc: menuDescription.getText(),(String)catDropdown.getSelectedItem(), url: imagefilepath));
        updateMenuList(mcdlist);
        selectedImage=null;
        return mcdlist;
    }
```

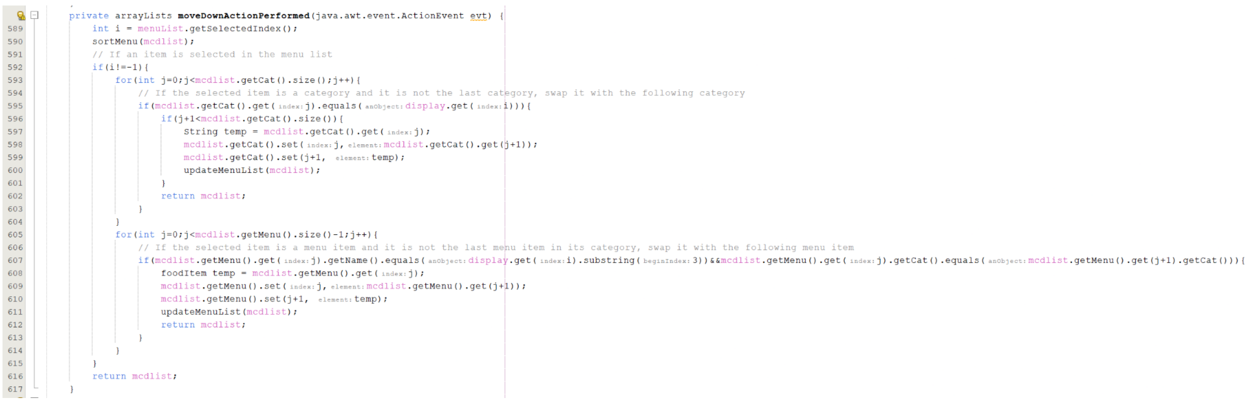


```
    private arrayLists moveUpActionPerformed(java.awt.event.ActionEvent evt) {
562     int i = menuList.getSelectedIndex();
563     sortMenu(mcdlist);
564     // If the selected item is not the first item in the list, move it up one position
565     if(i>0){
566         for(int j=0;j<mcdlist.getCat().size();j++){
567             // If the selected item is a category and it is not the first category, swap it with the preceding category
568             if(mcdlist.getCat().get( index:j).equals( anObject:display.get( index:i))){
569                 String temp = mcdlist.getCat().get(j-1);
570                 mcdlist.getCat().set(j-1, element:mcdlist.getCat().get( index:j));
571                 mcdlist.getCat().set( index:j,  element:temp);
572                 updateMenuList(mcdlist);
573                 return mcdlist;
574             }
575         }
576         for(int j=1;j<mcdlist.getMenu().size();j++){
577             // If the selected item is a menu item and it is not the first menu item in its category, swap it with the preceding menu item
578             if(mcdlist.getMenu().get( index:j).getName().equals( anObject:display.get( index:i).substring( beginIndex:3))&&mcdlist.getMenu().get( index:
579                 foodItem temp = mcdlist.getMenu().get(j-1);
580                 mcdlist.getMenu().set(j-1, element:mcdlist.getMenu().get( index:j));
581                 mcdlist.getMenu().set( index:j,  element:temp);
582                 updateMenuList(mcdlist);
583             }
584         }
585     }
586     return mcdlist;
587 }
```

Line 578 Continued:

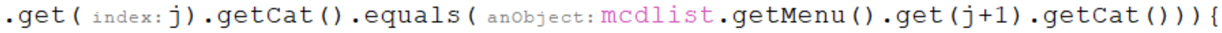


```
menu item
().get( index:j).getCat().equals( anObject:mcdlist.getMenu().get(j-1).getCat())){
```

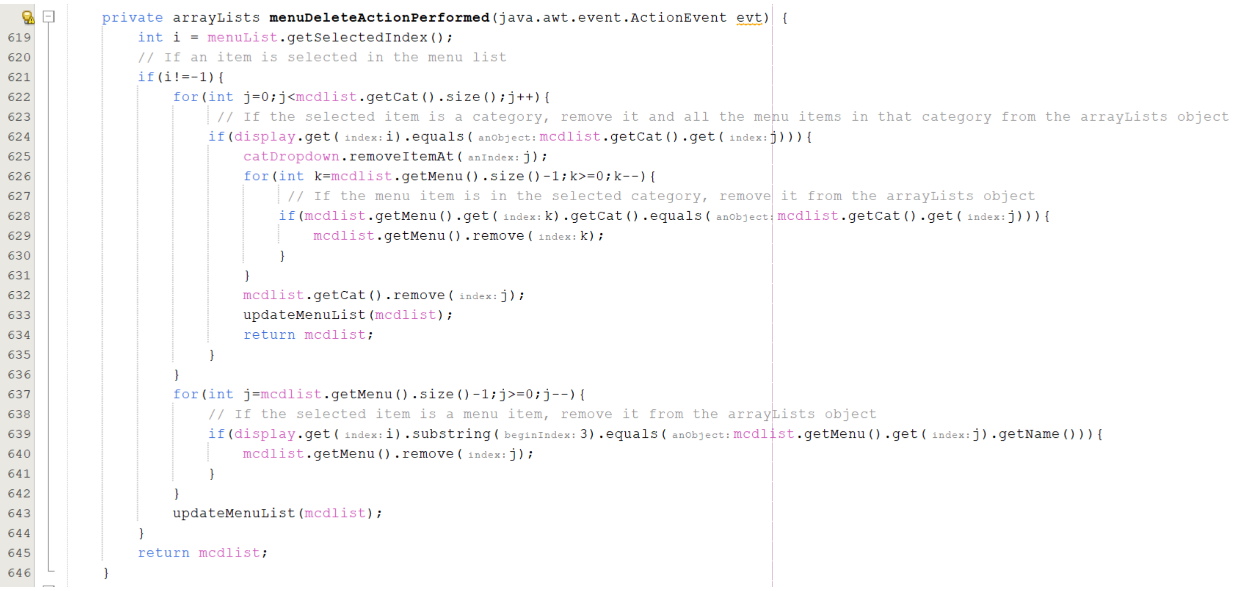


```
    private arrayLists moveDownActionPerformed(java.awt.event.ActionEvent evt) {
589         int i = menuList.getSelectedIndex();
590         sortMenu(mcdlist);
591         // If an item is selected in the menu list
592         if(i!=-1){
593             for(int j=0;j<mcdlist.getCat().size();j++){
594                 // If the selected item is a category and it is not the last category, swap it with the following category
595                 if(mcdlist.getCat().get( index: j).equals( anObject: display.get( index: i))){
596                     if(j+1<mcdlist.getCat().size()){
597                         String temp = mcdlist.getCat().get( index: j);
598                         mcdlist.getCat().set( index: j, element: mcdlist.getCat().get(j+1));
599                         mcdlist.getCat().set(j+1,  element: temp);
600                         updateMenuList(mcdlist);
601                     }
602                     return mcdlist;
603                 }
604             }
605             for(int j=0;j<mcdlist.getMenu().size()-1;j++){
606                 // If the selected item is a menu item and it is not the last menu item in its category, swap it with the following menu item
607                 if(mcdlist.getMenu().get( index: j).getName().equals( anObject: display.get( index: i).substring( beginIndex: 3))&&mcdlist.getMenu().get( index: j).getCat().equals( anObject: mcdlist.getMenu().get(j+1).getCat())){
608                     foodItem temp = mcdlist.getMenu().get( index: j);
609                     mcdlist.getMenu().set( index: j, element: mcdlist.getMenu().get(j+1));
610                     mcdlist.getMenu().set(j+1,  element: temp);
611                     updateMenuList(mcdlist);
612                     return mcdlist;
613                 }
614             }
615         }
616         return mcdlist;
617     }
```

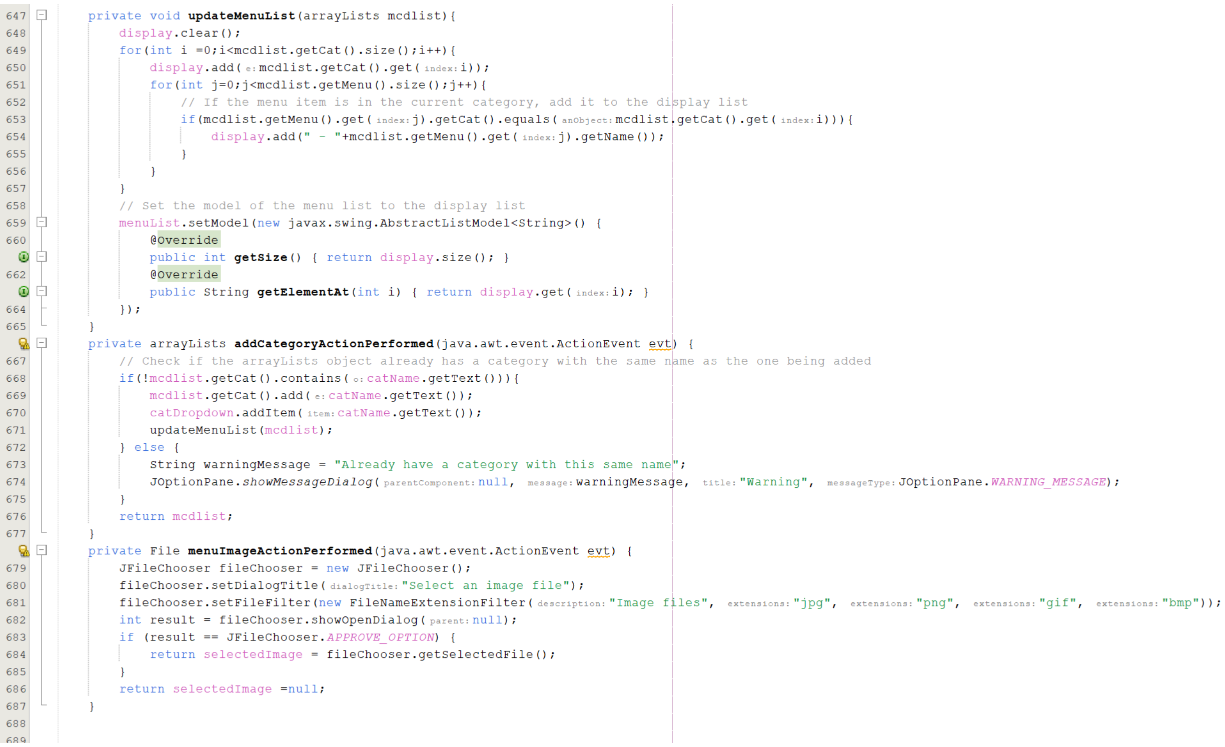
Line 607 Continued:



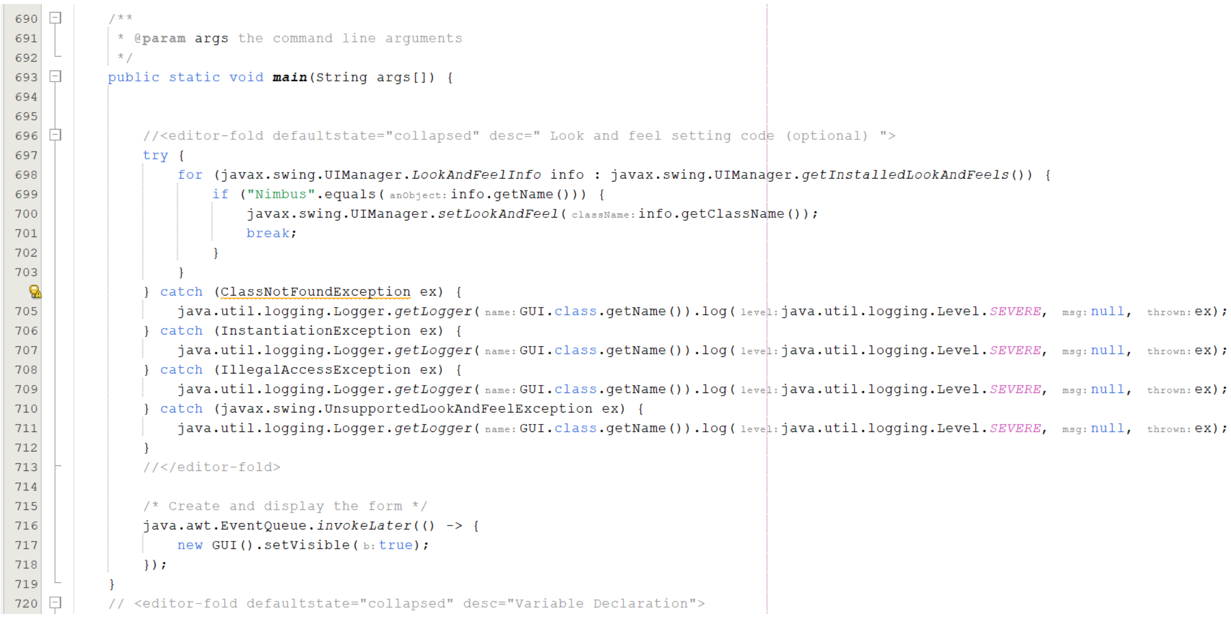
```
.get( index: j).getCat().equals( anObject: mcdlist.getMenu().get(j+1).getCat())){
```



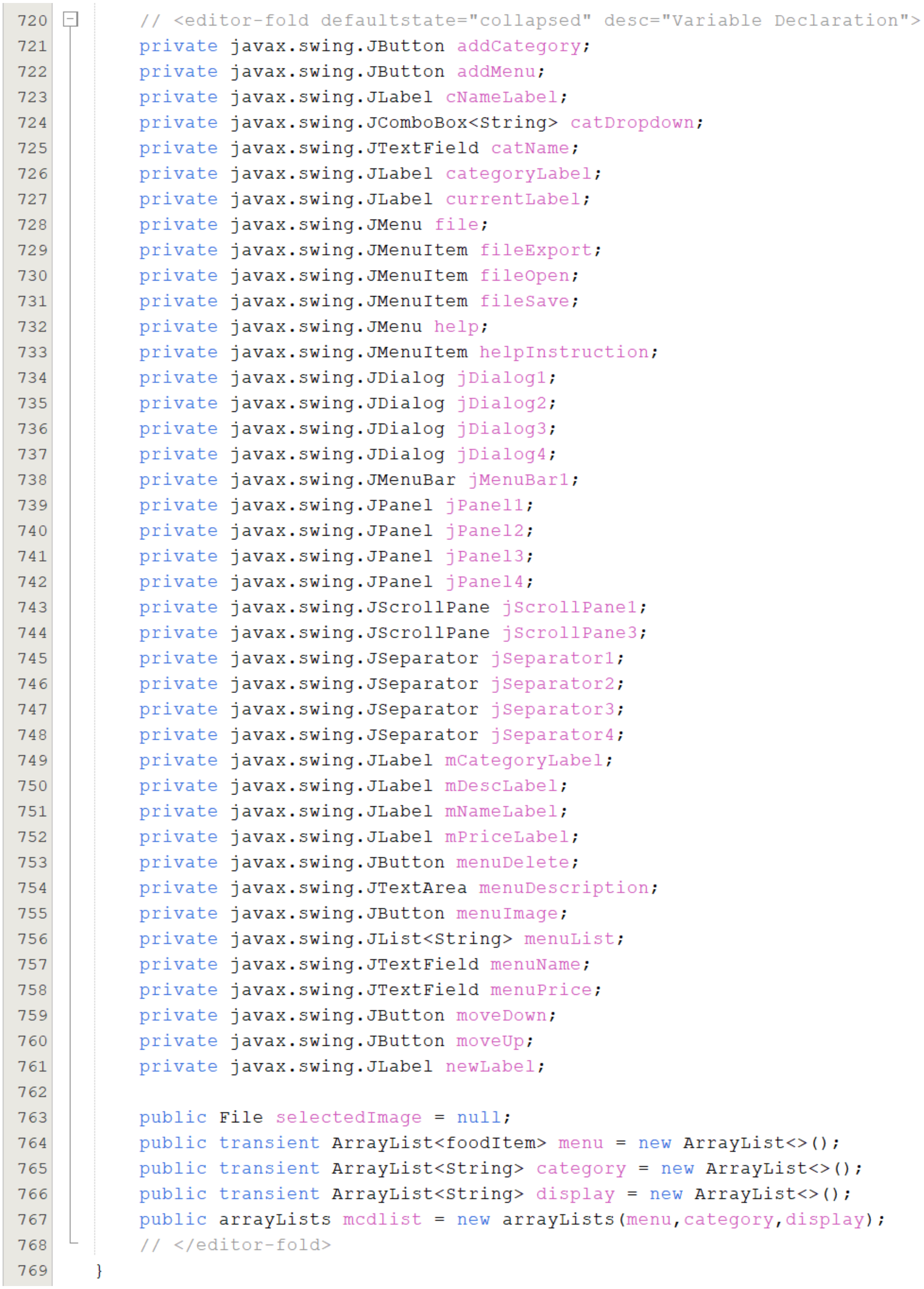
```java
    private arrayLists menuDeleteActionPerformed(java.awt.event.ActionEvent evt) {
        int i = menuList.getSelectedIndex();
        // If an item is selected in the menu list
        if(i!=-1){
            for(int j=0;j<mcdlist.getCat().size();j++){
                // If the selected item is a category, remove it and all the menu items in that category from the arrayLists object
                if(display.get( index:i).equals( anObject:mcdlist.getCat().get( index:j))){
                    catDropdown.removeItemAt( anIndex:j);
                    for(int k=mcdlist.getMenu().size()-1;k>=0;k--){
                        // If the menu item is in the selected category, remove it from the arrayLists object
                        if(mcdlist.getMenu().get( index:k).getCat().equals( anObject:mcdlist.getCat().get( index:j))){
                            mcdlist.getMenu().remove( index:k);
                        }
                    }
                    mcdlist.getCat().remove( index:j);
                    updateMenuList(mcdlist);
                    return mcdlist;
                }
            }
            for(int j=mcdlist.getMenu().size()-1;j>=0;j--){
                // If the selected item is a menu item, remove it from the arrayLists object
                if(display.get( index:i).substring( beginIndex:3).equals( anObject:mcdlist.getMenu().get( index:j).getName())){
                    mcdlist.getMenu().remove( index:j);
                }
            }
            updateMenuList(mcdlist);
        }
        return mcdlist;
    }
```



```java
    private void updateMenuList(arrayLists mcdlist){
        display.clear();
        for(int i =0;i<mcdlist.getCat().size();i++){
            display.add( e:mcdlist.getCat().get( index:i));
            for(int j=0;j<mcdlist.getMenu().size();j++){
                // If the menu item is in the current category, add it to the display list
                if(mcdlist.getMenu().get( index:j).getCat().equals( anObject:mcdlist.getCat().get( index:i))){
                    display.add(" - "+mcdlist.getMenu().get( index:j).getName());
                }
            }
        }
        // Set the model of the menu list to the display list
        menuList.setModel(new javax.swing.AbstractListModel<String>() {
            @Override
            public int getSize() { return display.size(); }
            @Override
            public String getElementAt(int i) { return display.get( index:i); }
        });
    }
    private arrayLists addCategoryActionPerformed(java.awt.event.ActionEvent evt) {
        // Check if the arrayLists object already has a category with the same name as the one being added
        if(!mcdlist.getCat().contains( o:catName.getText())){
            mcdlist.getCat().add( e:catName.getText());
            catDropdown.addItem( item:catName.getText());
            updateMenuList(mcdlist);
        } else {
            String warningMessage = "Already have a category with this same name";
            JOptionPane.showMessageDialog( parentComponent:null,  message:warningMessage,  title:"Warning",  messageType:JOptionPane.WARNING_MESSAGE);
        }
        return mcdlist;
    }
    private File menuImageActionPerformed(java.awt.event.ActionEvent evt) {
        JFileChooser fileChooser = new JFileChooser();
        fileChooser.setDialogTitle( dialogTitle:"Select an image file");
        fileChooser.setFileFilter(new FileNameExtensionFilter( description:"Image files",  extensions:"jpg",  extensions:"png",  extensions:"gif",  extensions:"bmp"));
        int result = fileChooser.showOpenDialog( parent:null);
        if (result == JFileChooser.APPROVE_OPTION) {
            return selectedImage = fileChooser.getSelectedFile();
        }
        return selectedImage =null;
    }
```



```java
/**
 * @param args the command line arguments
 */
public static void main(String args[]) {


    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Nimbus".equals( anObject: info.getName())) {
                javax.swing.UIManager.setLookAndFeel( className: info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger( name: GUI.class.getName()).log( level: java.util.logging.Level.SEVERE,  msg: null,  thrown: ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger( name: GUI.class.getName()).log( level: java.util.logging.Level.SEVERE,  msg: null,  thrown: ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger( name: GUI.class.getName()).log( level: java.util.logging.Level.SEVERE,  msg: null,  thrown: ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger( name: GUI.class.getName()).log( level: java.util.logging.Level.SEVERE,  msg: null,  thrown: ex);
    }
    //</editor-fold>

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(() -> {
        new GUI().setVisible( b: true);
    });
}
// <editor-fold defaultstate="collapsed" desc="Variable Declaration">
```



```
    // <editor-fold defaultstate="collapsed" desc="Variable Declaration">
    private javax.swing.JButton addCategory;
    private javax.swing.JButton addMenu;
    private javax.swing.JLabel cNameLabel;
    private javax.swing.JComboBox<String> catDropdown;
    private javax.swing.JTextField catName;
    private javax.swing.JLabel categoryLabel;
    private javax.swing.JLabel currentLabel;
    private javax.swing.JMenu file;
    private javax.swing.JMenuItem fileExport;
    private javax.swing.JMenuItem fileOpen;
    private javax.swing.JMenuItem fileSave;
    private javax.swing.JMenu help;
    private javax.swing.JMenuItem helpInstruction;
    private javax.swing.JDialog jDialog1;
    private javax.swing.JDialog jDialog2;
    private javax.swing.JDialog jDialog3;
    private javax.swing.JDialog jDialog4;
    private javax.swing.JMenuBar jMenuBar1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel3;
    private javax.swing.JPanel jPanel4;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane3;
    private javax.swing.JSeparator jSeparator1;
    private javax.swing.JSeparator jSeparator2;
    private javax.swing.JSeparator jSeparator3;
    private javax.swing.JSeparator jSeparator4;
    private javax.swing.JLabel mCategoryLabel;
    private javax.swing.JLabel mDescLabel;
    private javax.swing.JLabel mNameLabel;
    private javax.swing.JLabel mPriceLabel;
    private javax.swing.JButton menuDelete;
    private javax.swing.JTextArea menuDescription;
    private javax.swing.JButton menuImage;
    private javax.swing.JList<String> menuList;
    private javax.swing.JTextField menuName;
    private javax.swing.JTextField menuPrice;
    private javax.swing.JButton moveDown;
    private javax.swing.JButton moveUp;
    private javax.swing.JLabel newLabel;

    public File selectedImage = null;
    public transient ArrayList<foodItem> menu = new ArrayList<>();
    public transient ArrayList<String> category = new ArrayList<>();
    public transient ArrayList<String> display = new ArrayList<>();
    public arrayLists mcdlist = new arrayLists(menu,category,display);
    // </editor-fold>
}
```