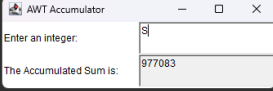


LAB 05

1. Swing components

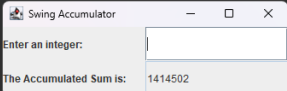
1.1. AWTAccumulator

```
1 package hust.soict.dsai.swing;
2
3 import java.awt.Frame;
4
5 public class AWTAccumulator extends Frame {
6     private TextField tfInput;
7     private TextField tfOutput;
8     private int sum = 0;
9
10    // Constructor to setup the GUI components and event handlers
11    public AWTAccumulator() {
12        setLayout(new GridLayout(2,2));
13        add(new Label("Enter an integer: "));
14        tfInput = new TextField(10);
15        add(tfInput);
16        tfInput.addActionListener(new TFInputListener());
17
18        add(new Label("The Accumulated Sum is: "));
19
20        tfOutput = new TextField(10);
21        tfOutput.setEditable(false);
22        add(tfOutput);
23
24        setTitle("AWT Accumulator");
25        setSize(350,120);
26        setVisible(true);
27    }
28
29    public static void main(String [] args) {
30        Frame frame = new AWTAccumulator();
31        frame.addWindowListener(new WindowAdapter() {
32            public void windowClosing(WindowEvent e) {
33                // Perform any necessary cleanup tasks here
34                System.exit(0); // Terminate the application
35            }
36        });
37    }
38
39    private class TFInputListener implements ActionListener {
```

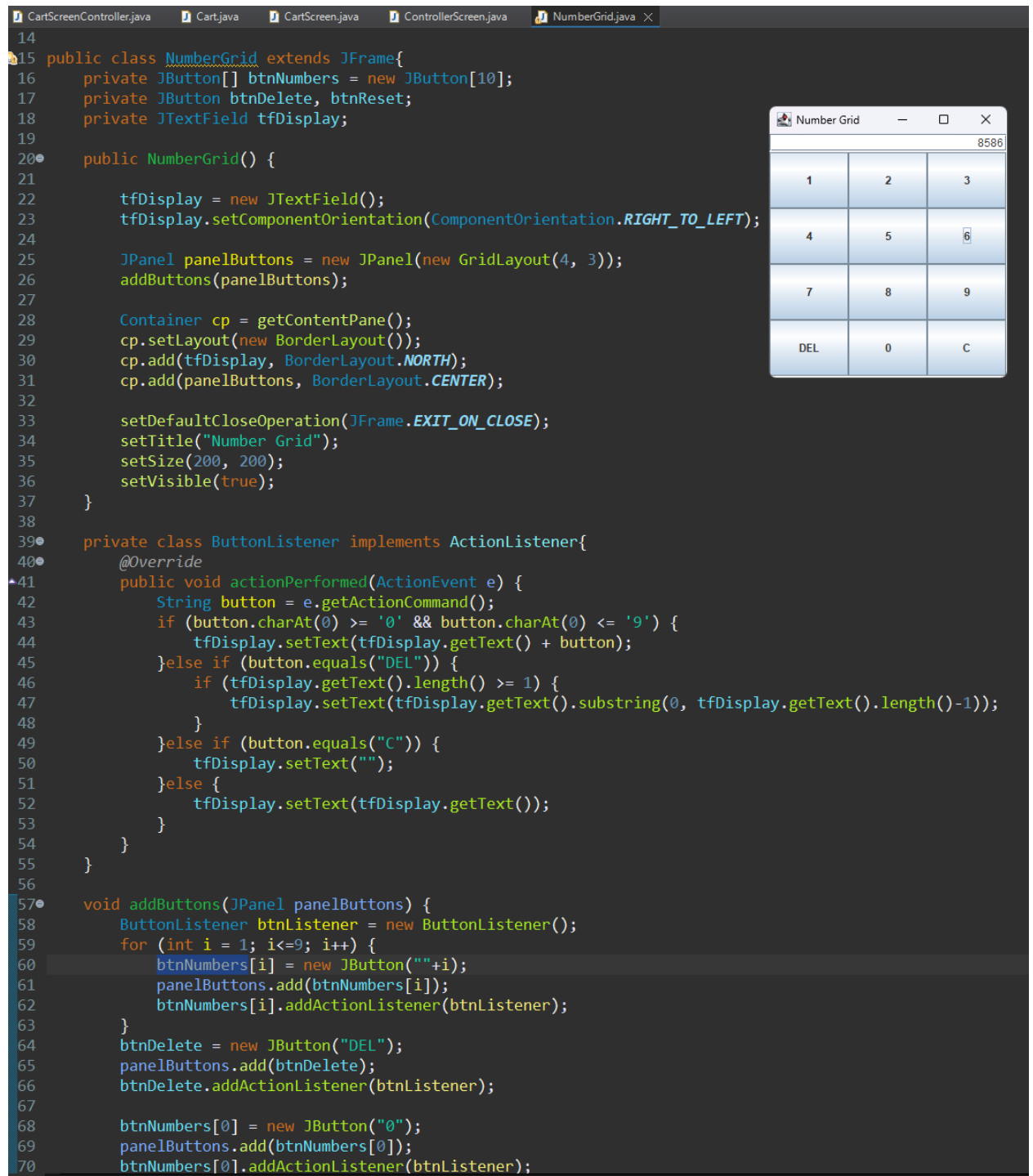


1.2. SwingAccumulator

```
1 package hust.soict.dsai.swing;
2
3 import java.awt.Container;
4
5 public class SwingAccumulator extends JFrame {
6     private JTextField tfInput;
7     private JTextField tfOutput;
8     private int sum = 0;
9
10    // Constructor to setup the GUI components and event handlers
11    public SwingAccumulator() {
12        Container cp = getContentPane();
13        cp.setLayout(new GridLayout(2,2));
14        cp.add(new JLabel("Enter an integer: "));
15        tfInput = new JTextField(10);
16        cp.add(tfInput);
17        tfInput.addActionListener(new TFInputListener());
18
19        cp.add(new JLabel("The Accumulated Sum is: "));
20
21        tfOutput = new JTextField(10);
22        tfOutput.setEditable(false);
23        cp.add(tfOutput);
24
25        setTitle("Swing Accumulator");
26        setSize(350,120);
27        setVisible(true);
28    }
29
30    public static void main(String [] args) {
31        new SwingAccumulator();
32    }
33
34    private class TFInputListener implements ActionListener {
35        @Override
36        public void actionPerformed(ActionEvent evt) {
37            int numberIn = Integer.parseInt(tfInput.getText());
38            sum += numberIn;
39            tfInput.setText("");
40            tfOutput.setText(sum + "");
41        }
42    }
```



2. Organizing Swing components with Layout Managers



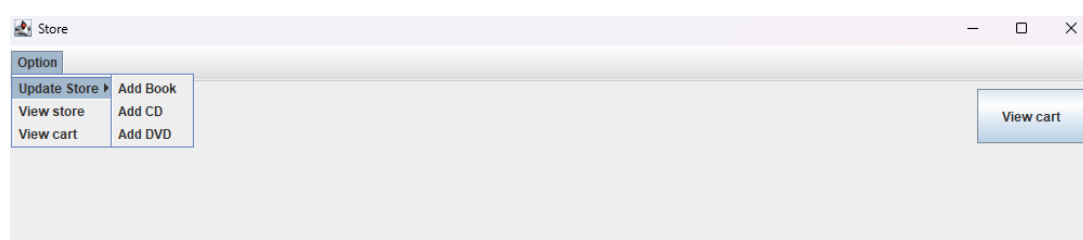
The screenshot shows an IDE with a Java file named `NumberGrid.java`. The code defines a `NumberGrid` class that extends `JFrame`. It includes a text field for displaying numbers, a grid of buttons for digits 0-9, a 'DEL' button, and a 'C' button. The code uses `GridLayout` for the button grid and `BorderLayout` for the main frame layout. A `ButtonListener` class implements `ActionListener` to handle button clicks, updating the text field or performing actions like deletion or clearing.

Next to the code is a preview of the application window titled "Number Grid". The window has a text field at the top displaying "8586". Below it is a 4x3 grid of buttons. The first three rows contain digits 1-9, and the last row contains "DEL", "0", and "C".

```
14
15 public class NumberGrid extends JFrame{
16     private JButton[] btnNumbers = new JButton[10];
17     private JButton btnDelete, btnReset;
18     private JTextField tfDisplay;
19
20     public NumberGrid() {
21
22         tfDisplay = new JTextField();
23         tfDisplay.setComponentOrientation(ComponentOrientation.RIGHT_TO_LEFT);
24
25         JPanel panelButtons = new JPanel(new GridLayout(4, 3));
26         addButtons(panelButtons);
27
28         Container cp = getContentPane();
29         cp.setLayout(new BorderLayout());
30         cp.add(tfDisplay, BorderLayout.NORTH);
31         cp.add(panelButtons, BorderLayout.CENTER);
32
33         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
34         setTitle("Number Grid");
35         setSize(200, 200);
36         setVisible(true);
37     }
38
39     private class ButtonListener implements ActionListener{
40         @Override
41         public void actionPerformed(ActionEvent e) {
42             String button = e.getActionCommand();
43             if (button.charAt(0) >= '0' && button.charAt(0) <= '9') {
44                 tfDisplay.setText(tfDisplay.getText() + button);
45             }else if (button.equals("DEL")) {
46                 if (tfDisplay.getText().length() >= 1) {
47                     tfDisplay.setText(tfDisplay.getText().substring(0, tfDisplay.getText().length()-1));
48                 }
49             }else if (button.equals("C")) {
50                 tfDisplay.setText("");
51             }else {
52                 tfDisplay.setText(tfDisplay.getText());
53             }
54         }
55     }
56
57     void addButtons(JPanel panelButtons) {
58         ButtonListener btnListener = new ButtonListener();
59         for (int i = 1; i<=9; i++) {
60             btnNumbers[i] = new JButton(""+i);
61             panelButtons.add(btnNumbers[i]);
62             btnNumbers[i].addActionListener(btnListener);
63         }
64         btnDelete = new JButton("DEL");
65         panelButtons.add(btnDelete);
66         btnDelete.addActionListener(btnListener);
67
68         btnNumbers[0] = new JButton("0");
69         panelButtons.add(btnNumbers[0]);
70         btnNumbers[0].addActionListener(btnListener);
71     }
72 }
```

3. Create a graphical user interface for AIMS with Swing

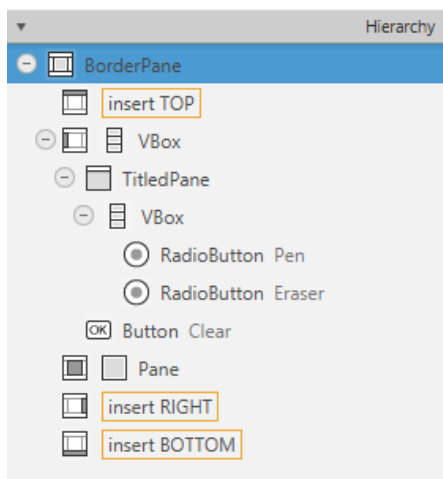
3.1. View Store Screen



3.2. Adding more user interaction

```
2      JButton addCartBtn = new JButton("Add to Cart");
3      addCartBtn.addActionListener(new ActionListener() {
4
5          @Override
6          public void actionPerformed(ActionEvent e) {
7              try {
8                  cart.addMedia(media);
9              } catch (LimitExceededException e1) {
10                 // TODO Auto-generated catch block
11                 e1.printStackTrace();
12             }
13         }
14     });
15     container.add(addCartBtn);
16
17
18     if (media instanceof Playable) {
19         JButton playBtn = new JButton("Play");
20         playBtn.addActionListener(new ActionListener() {
21             @Override
22             public void actionPerformed(ActionEvent e) {
23                 JDialog playDialog = new JDialog();
24                 JPanel mainGui = new JPanel(new BorderLayout());
25                 mainGui.setBorder(new EmptyBorder(20, 20, 20, 20));
26
27                 // Display Playing Message
28                 mainGui.add(new JLabel("Playing... " + media.getTitle(), BorderLayout.CENTER);
29                 System.out.println(media.getTitle());
30                 // Close Button
31                 JPanel buttonPanel = new JPanel(new FlowLayout());
32                 JButton close = new JButton("Stop");
33                 close.addActionListener(ev -> {
34                     playDialog.setVisible(false);
35                     System.out.println("Stopped playing.");
36                 });
37                 buttonPanel.add(close);
38                 mainGui.add(buttonPanel, BorderLayout.SOUTH);
39
40                 playDialog.setContentPane(mainGui);
41                 playDialog.setLocationRelativeTo(playBtn);
42                 playDialog.pack();
43
44                 // Show Dialog
45                 playDialog.setVisible(true);
46             }
47         });
48         container.add(playBtn);
49     }
```

4. JavaFX API



Inspector

Properties : BorderLayout

Layout : BorderLayout

Internal

Padding

8

>

8

8

8

Size

Min Width

USE_PREF_SIZE

Min Height

USE_PREF_SIZE

Pref Width

640

Pref Height

480

Max Width

USE_PREF_SIZE

Max Height

USE_PREF_SIZE

Width

640

Height

480

Position

Layout X

0

Layout Y

0

Transforms

Rotate

0

Rotation Axis

X

0

Y

0

Z

1

Scale X

1

Scale Y

1

Scale Z

1

Translate X

0

Translate Y

0

Translate Z

0

Bounds

Layout Bounds

0,0 640x480

Bounds In Local

0,0 640x480

Bounds In Parent

0,0 640x480

Inspector

Properties : Pane

Layout : Pane

Code : Pane

Identity

fx:id

DragDrop

On Drag Detected

#

On Drag Done

#

On Drag Dropped

#

Layout : VBox

Border Pane Constraints

Alignment

Margin >

Internal

Padding >

Spacing

Specific

Fill Width ☒

Size

Min Width

Min Height

Pref Width

Pref Height

Max Width

Max Height

Width

Height

Position

BorderPane

insert TOP

VBox

TitledPane

VBox

RadioButton Pen

RadioButton Eraser

OK Button Clear

Pane

insert RIGHT

insert BOTTOM

Style

-fx-background-color

Style Class

Stylesheets

Id

Extras

On Context Menu Requested

#

On Mouse Clicked

#

On Mouse Dragged

#

On Mouse Entered

#

On Mouse Exited

BorderPane

insert TOP

VBox

TitledPane

VBox

RadioButton Pen

RadioButton Eraser

OK Button Clear

Pane

insert RIGHT

insert BOTTOM

Properties : Button

Text

Text

Font

Text Fill

Wrap Text ☐

Text Alignment

Text Overrun

Ellipsis String

Underline ☐

Line Spacing

Layout : Button

VBox Constraints

Vgrow

Margin ☐ 0 ☐ 0 ☐ 0 ☐ 0

Internal

Padding ☐ 0 ☐ 0 ☐ 0 ☐ 0

Size

Min Width

Min Height

Pref Width

Pref Height

Max Width

Max Height

Width

Height

Code : Button

On Action

DragDrop

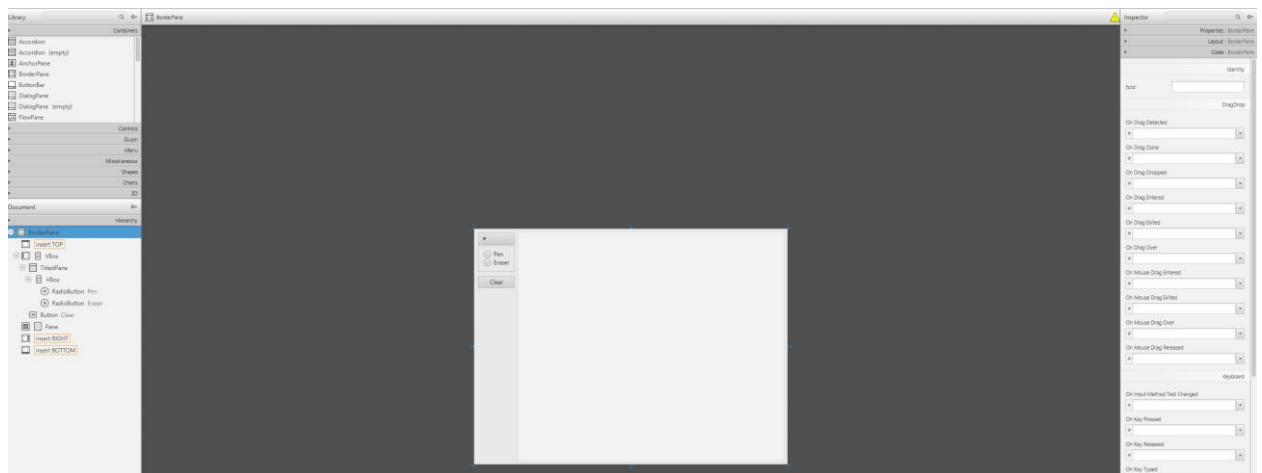
On Drag Detected

On Drag Done

On Drag Dropped

On Drag Entered

On Drag Exited



```

1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <?import javafx.geometry.Insets?>
4 <?import javafx.scene.control.Button?>
5 <?import javafx.scene.control.RadioButton?>
6 <?import javafx.scene.control.ToggleGroup?>
7 <?import javafx.scene.layout.BorderPane?>
8 <?import javafx.scene.layout.Pane?>
9 <?import javafx.scene.layout.VBox?>
10
11 Bind to grammar/schema...
12 <BorderPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="480.0" prefWidth="640.0" xmlns="http://javafx.com/javafx/8" >
13   <left>
14     <VBox maxHeight="1.7976931348623157E308" spacing="8.0" BorderPane.alignment="CENTER">
15       <BorderPane.margin>
16         <Insets right="8.0" />
17       </BorderPane.margin>
18       <children>
19         <TitledPane>
20           <content>
21             <VBox>
22               <children>
23                 <RadioButton mnemonicParsing="false" onAction="#chooseOption" text="Pen">
24                   <toggleGroup>
25                     <ToggleGroup fx:id="identity" />
26                   </toggleGroup>
27                 </RadioButton>
28                 <RadioButton mnemonicParsing="false" onAction="#chooseOption" text="Eraser" toggleGroup="identity" />
29               </children>
30             </VBox>
31           </content>
32         </TitledPane>
33         <Button maxWidth="1.7976931348623157E308" mnemonicParsing="false" onAction="#clearButtonPressed" text="Clear" />
34       </children>
35     </VBox>
36   </left>
37   <center>
38     <Pane fx:id="drawingAreaPane" onMouseDragged="#drawingAreaMouseDragged" style="-fx-background-color: white;" BorderPane.alignment="CENTER" />
39   </center>
40   <padding>
41     <Insets bottom="8.0" left="8.0" right="8.0" top="8.0" />
42   </padding>
43 </BorderPane>
44

```

```

1 package hust.soict.dsai.javafx;
2
3 import javafx.event.ActionEvent;
4
5 public class PainterController {
6
7     private int color;
8
9     @FXML
10     private ToggleGroup identity;
11
12     @FXML
13     void chooseOption(ActionEvent event) {
14         String button = ((RadioButton)event.getSource()).getText();
15         if (button.equals("Pen")) {
16             System.out.println("1");
17             color = 1;
18         } else {
19             System.out.println("0");
20             color = 0;
21         }
22     }
23
24     @FXML
25     private Pane drawingAreaPane;
26
27     @FXML
28     void clearButtonPressed(ActionEvent event) {
29         drawingAreaPane.getChildren().clear();
30     }
31
32     @FXML
33     void drawingAreaMouseDragged(MouseEvent event) {
34         if (color == 1 && event.getX() >= 0) {
35             Circle newCircle = new Circle(event.getX(), event.getY(), 4.0, Color.BLACK);
36             drawingAreaPane.getChildren().add(newCircle);
37         } else if (color == 0 && event.getX() >= 0) {
38             Circle newCircle = new Circle(event.getX(), event.getY(), 4.0, Color.WHITE);
39             drawingAreaPane.getChildren().add(newCircle);
40         }
41     }
42 }
43

```



```
1 package hust.soict.dsai.javafx;
2
3 import javafx.application.Application;
4
5
6
7
8
9 public class Painter extends Application{
10
11     @Override
12     public void start(Stage stage) throws Exception{
13         Parent root = FXMLLoader.Load(getClass().getResource("/hust/soict/dsai/javafx/Painter.fxml"));
14
15         Scene scene = new Scene(root);
16         stage.setTitle("Painter");
17         stage.setScene(scene);
18         stage.show();
19     }
20
21     public static void main(String[] args) {
22         launch(args);
23     }
24 }
25
```

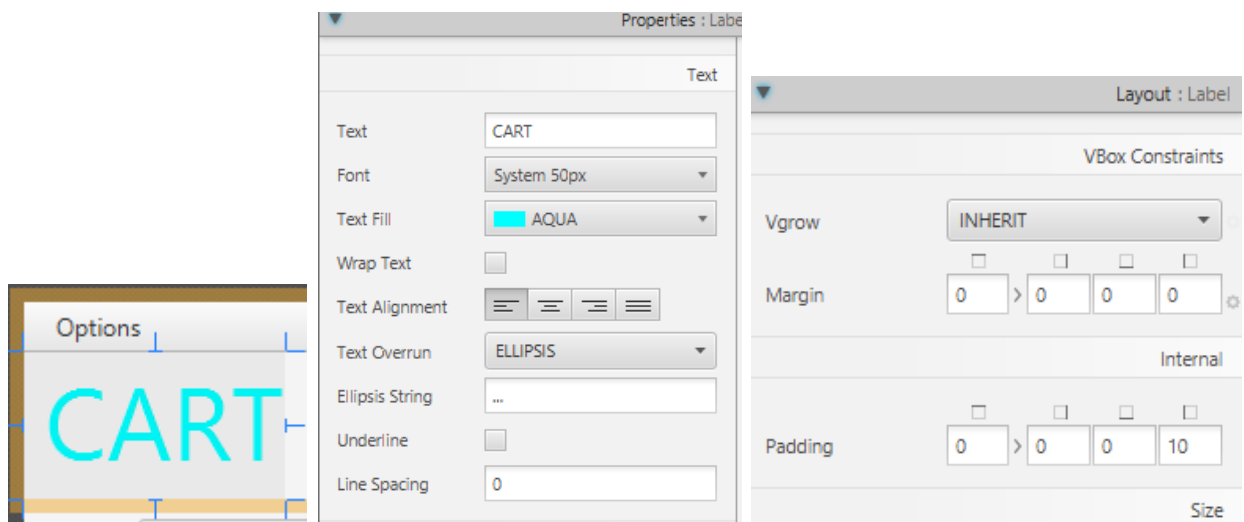
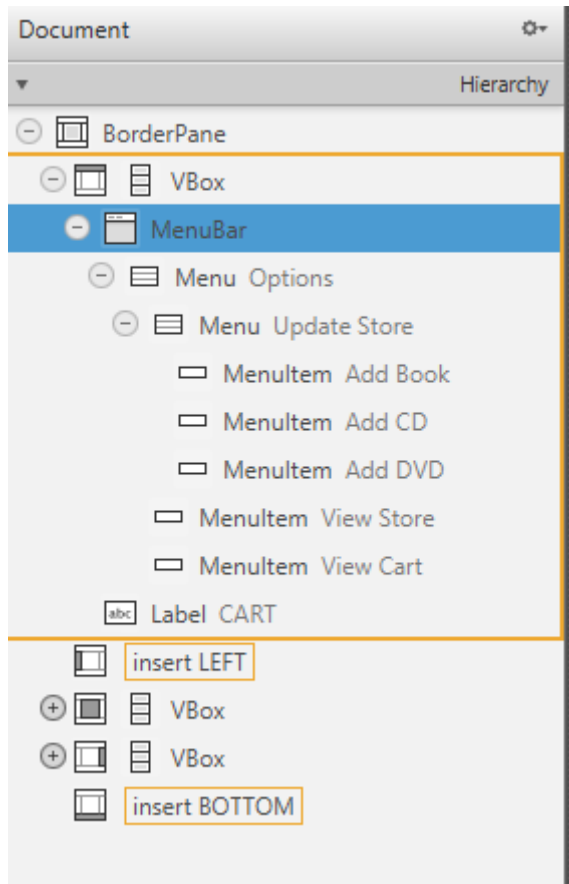


Handle Erase Function

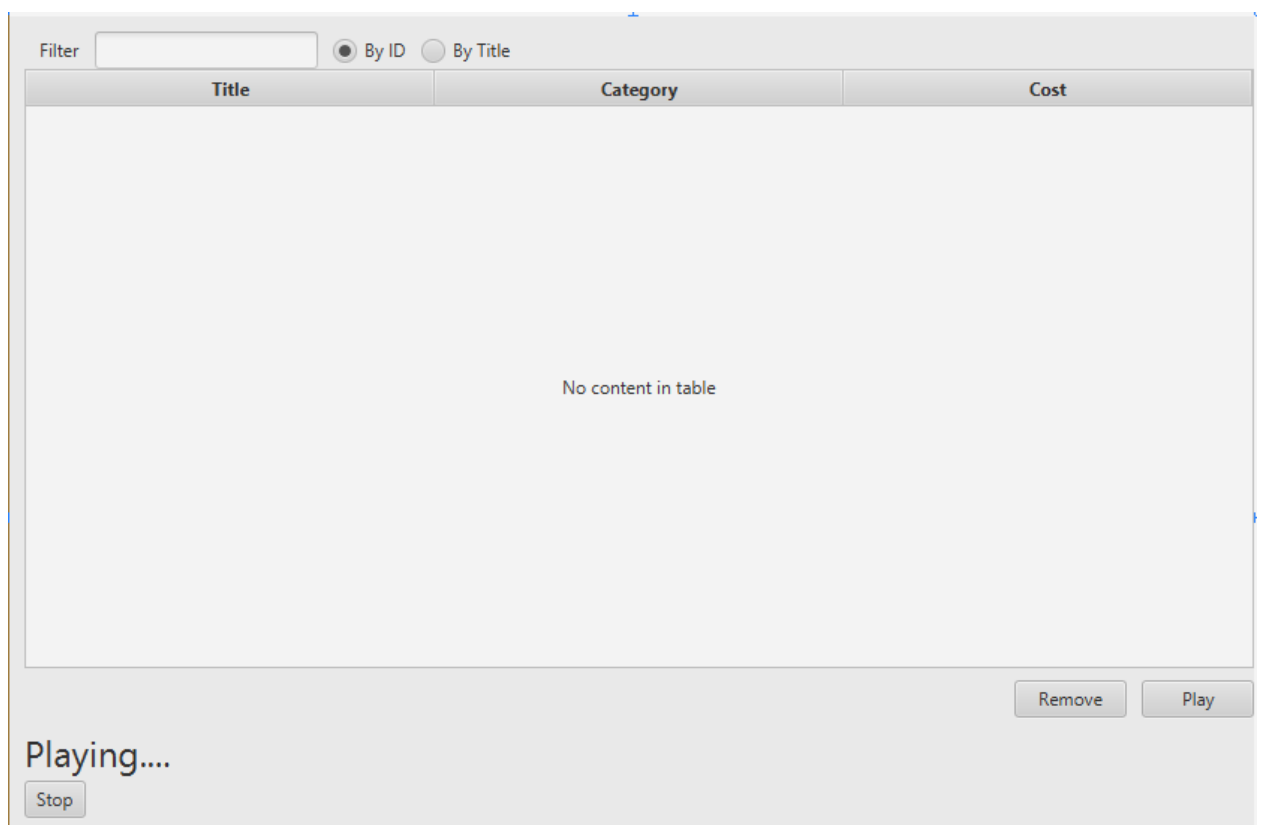
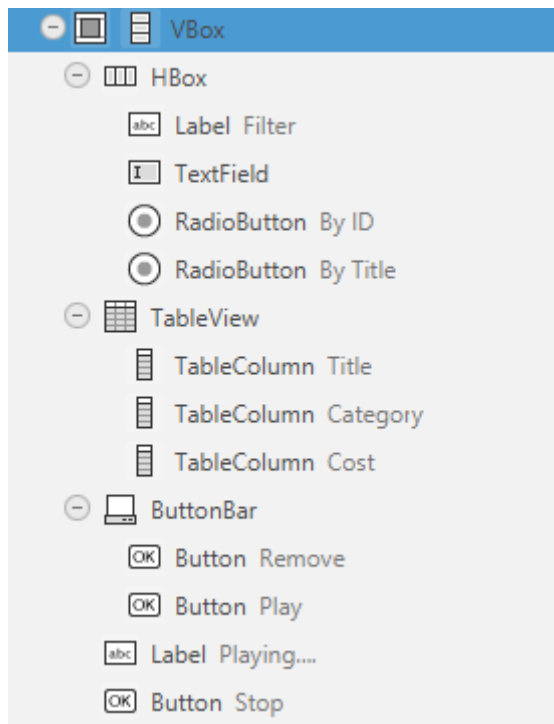
```
@FXML
void drawingAreaMouseDragged(MouseEvent event) {
    if (color == 1 && event.getX() >= 0) {
        Circle newCircle = new Circle(event.getX(), event.getY(), 4.0, Color.BLACK);
        drawingAreaPane.getChildren().add(newCircle);
    } else if (color == 0 && event.getX() >= 0) {
        Circle newCircle = new Circle(event.getX(), event.getY(), 4.0, Color.WHITE);
        drawingAreaPane.getChildren().add(newCircle);
    }
}
```

5. Setting up the View Cart Screen with ScreenBuilder

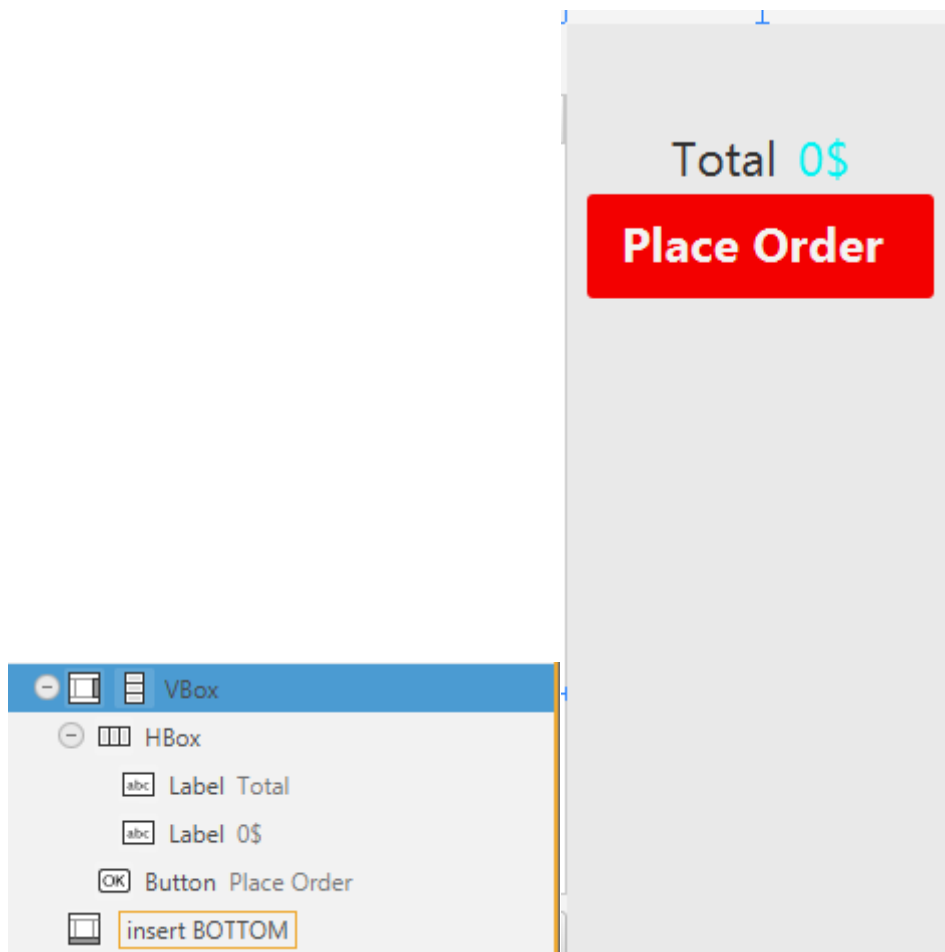
5.2. Setting up the TOP area



5.3. Setting up the CENTER area



5.4. Setting up the RIGHT area



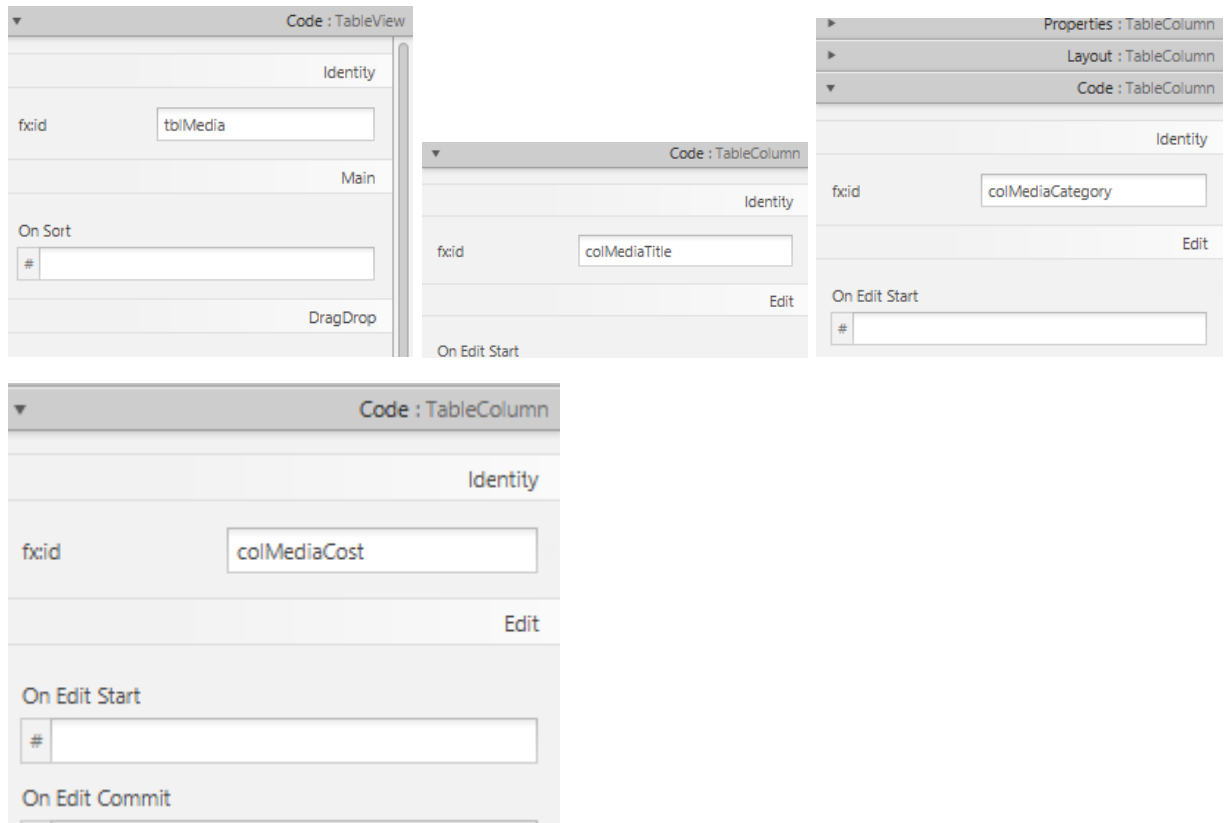
6. Integrating JavaFX into Swing application – The JFXPanel class

```

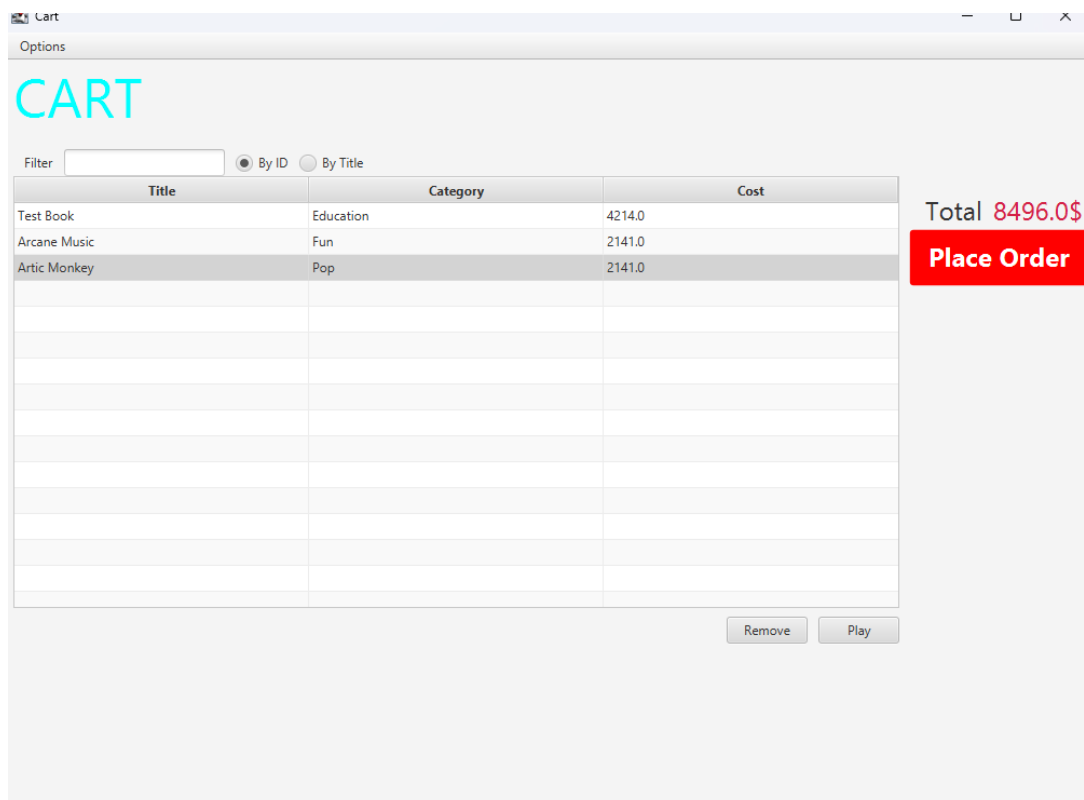
1 package hust.soict.dsai.aims.screen;
2
3 import java.awt.Dimension;
4
15 public class CartScreen extends JFrame{
16
17     private Cart cart;
18     private ControllerScreen controllerScreen;
19
20     public CartScreen(Cart cart, ControllerScreen c) {
21         super();
22
23         this.cart = cart;
24
25         JFXPanel fxPanel = new JFXPanel();
26         this.add(fxPanel);
27         this.setTitle("Cart");
28         this.setVisible(true);
29         this.setPreferredSize(new Dimension(1024, 768));
30         pack();
31         Platform.runLater(new Runnable() {
32
33             @Override
34             public void run() {
35                 try {
36                     FXMLLoader loader = new FXMLLoader(getClass()
37                         .getResource("/hust/soict/dsai/aims/screen/cart.fxml"));
38                     CartScreenController controller = new CartScreenController(cart,c);
39                     loader.setController(controller);
40                     Parent root = loader.load();
41                     fxPanel.setScene(new Scene(root,1024,768));
42                 } catch (IOException e) {
43                     e.printStackTrace();
44                 }
45             }
46         });
47     }
48 }

```

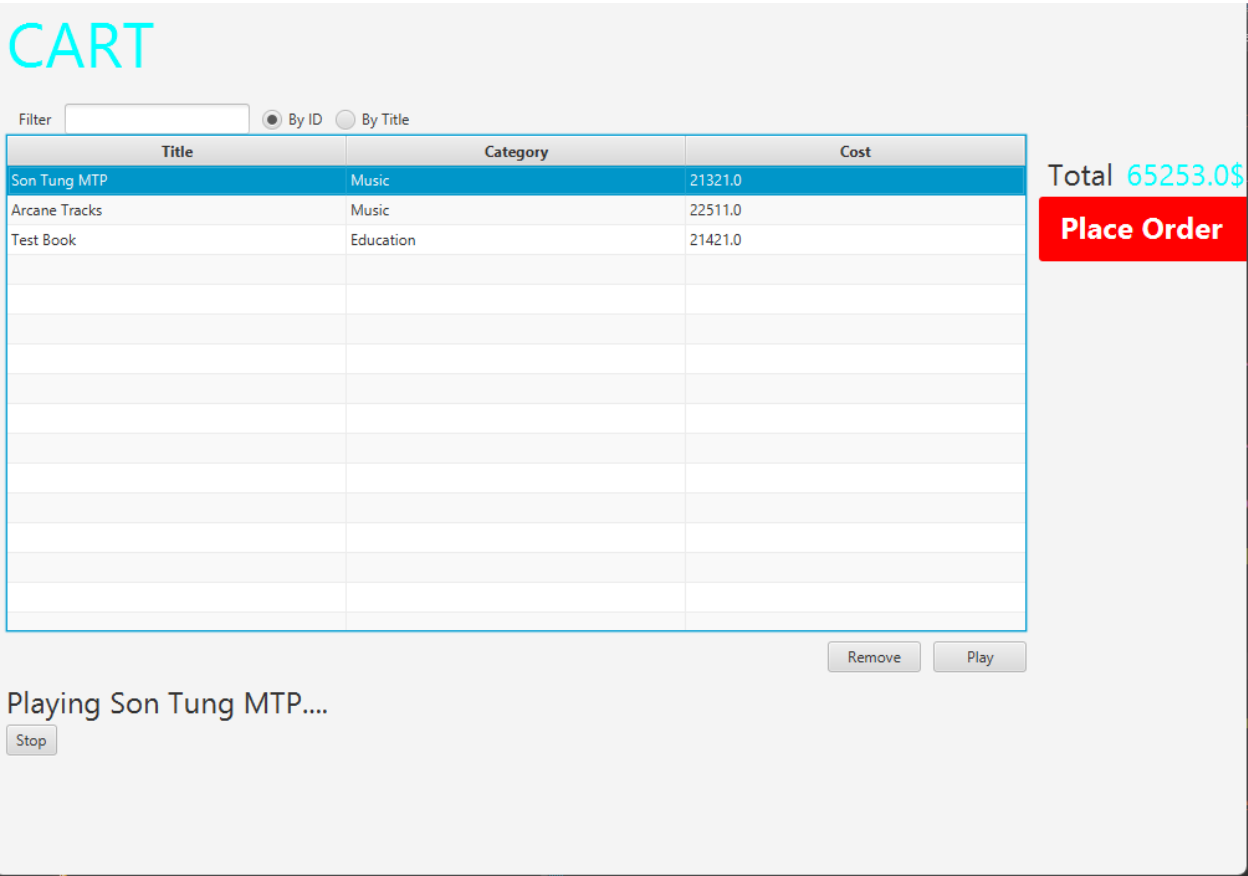
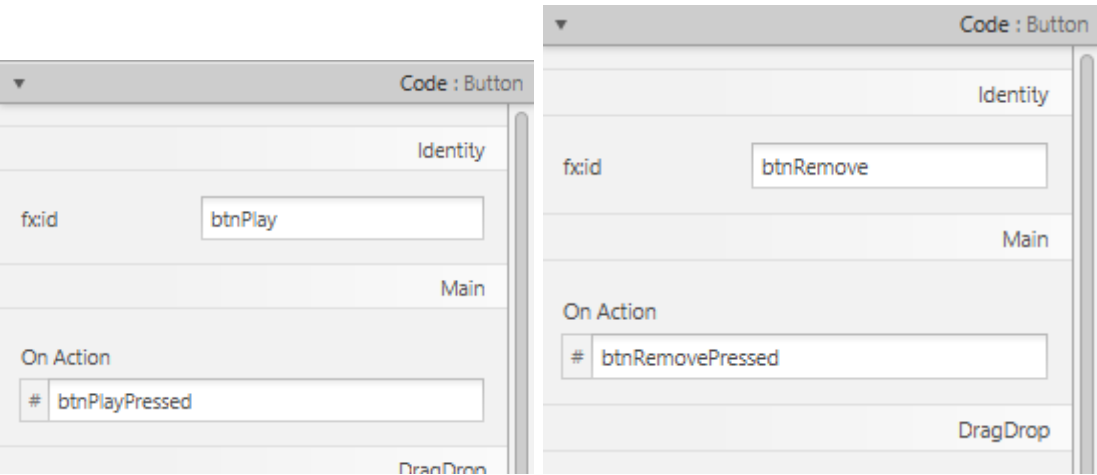
7. View the items in cart – JavaFX's data-driven UI



- Update itemsOrdered to ObservableList



8. Updating buttons based on selected item in TableView –
ChangeListener

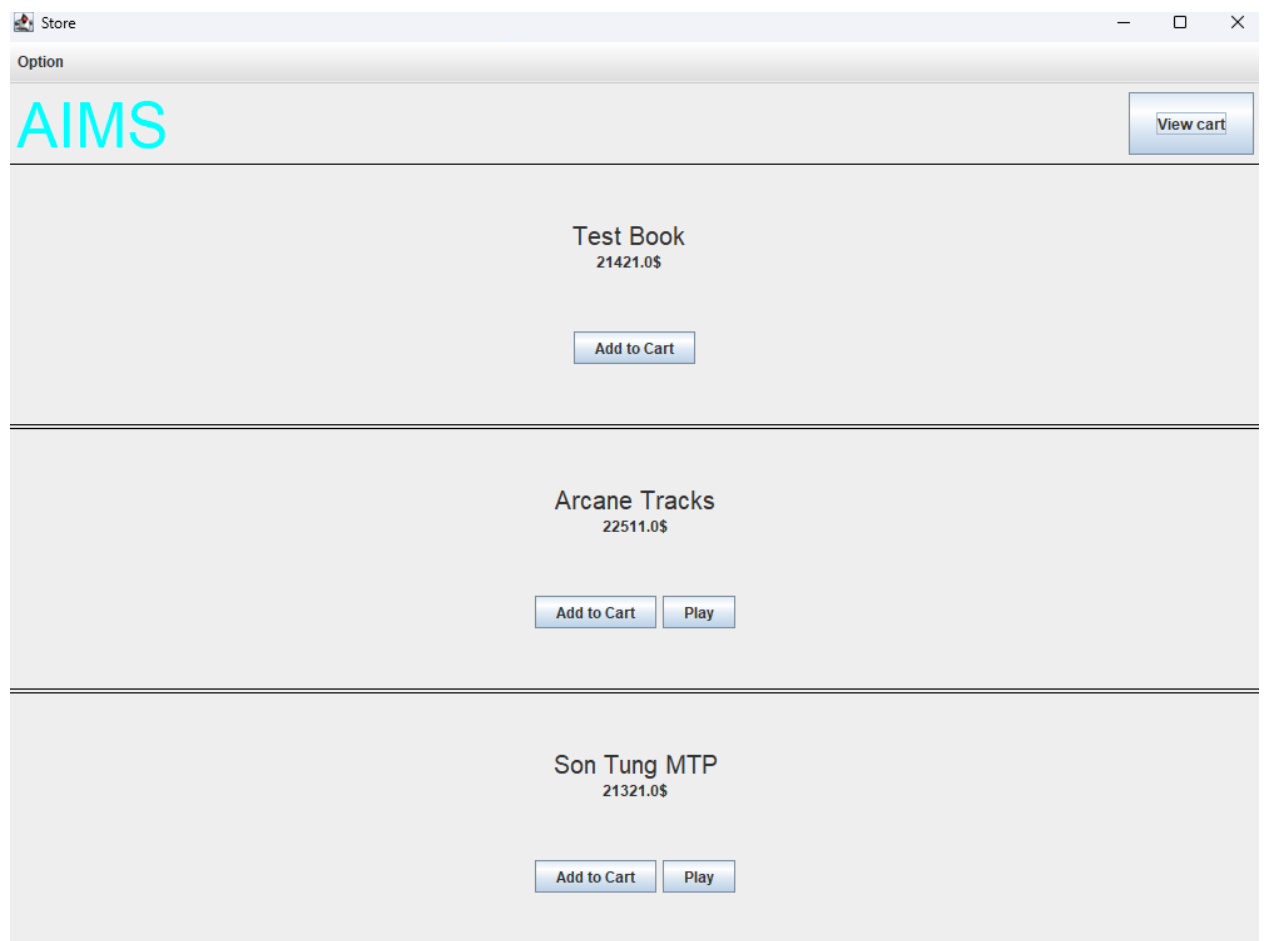



```

114     tfFilter.textProperty().addListener(new ChangeListener<String>() {
115
116         @Override
117         public void changed(ObservableValue<? extends String> observable, String oldValue, String newValue) {
118             showFilterMedia(newValue);
119         }
120     });
121
122     tblMedia.getSelectionModel().selectedItemProperty().addListener(
87     void showFilterMedia(String searchString) {
88         if(searchString.isEmpty()) {
89             tblMedia.setItems(this.cart.getItemsOrdered());
90         } else {
91             if(radioBtnFilterId.isSelected()) {
92                 tblMedia.setItems(new FilteredList<Media>(this.cart.getItemsOrdered(), item-> item.getId()==Integer.parseInt(searchString)));
93             } else
94                 tblMedia.setItems(new FilteredList<Media>(this.cart.getItemsOrdered(), item-> item.getTitle().contains(searchString)));
95         }
96     }
97

```

11. Complete the Aims GUI application



Cart

Options

CART

Filter

☐ By ID ☒ By Title

Title	Category	Cost
Test Book	Education	21421.0
Arcane Tracks	Music	22511.0
Son Tung MTP	Music	21321.0

Total 65253.0

Order

Remove

Play

Add Book

Add Book

Option

AIMS

View cart

title

category

cost

Authors(Names are separated by a comma)

add

Add CD

Add CD

Option

AIMS

View cart

title

category

cost

artist

List track (each track separated by a comma Ex: track-length)

add

Add DVD

Add DVD

Option

AIMS

View cart

title

category

cost

director

length

add

Filter by Title

Cart

Options

CART

Filter

Test

☐ By ID
 ☒ By Title

Title	Category	Cost
Test Book	Education	21421.0

Total 65253.0

Order

Filter by Id

Cart

Options

CART

Filter ☒ By ID ☐ By Title

Title	Category	Cost
Son Tung MTP	Music	21321.0

Total 65253.0

Order

AddItemToStoreScreen

```

1 package hust.soict.dsai.aims.screen;
2
3 import java.awt.BorderLayout;
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29 public class AddItemToStoreScreen extends JFrame {
30     protected Store store;
31     protected Cart cart;
32     protected JTextField title;
33     protected JTextField category;
34     protected JTextField cost;
35     protected JButton addBtn;
36     protected ControllerScreen c;
37     protected JPanel center;
38     protected int numberInput=3;
39     JMenuBar createMenuBar() {
40         JMenu menu = new JMenu("Option");
41
42         JMenu smUpdateStore = new JMenu("Update Store");
43         JMenuItem addBookMenu= new JMenuItem("Add Book");
44         addBookMenu.addActionListener(e->{
45             c.showAddBookScreen();
46         });
47         smUpdateStore.add(addBookMenu);
48         JMenuItem addCDMenu=new JMenuItem("Add CD");
49         addCDMenu.addActionListener(e->{
50             c.showAddCDCreen();
51         });
52         smUpdateStore.add(addCDMenu);
53         JMenuItem addDVDMenu= new JMenuItem("Add DVD");
54         addDVDMenu.addActionListener(e->{
55             c.showAddDVDScreen();
56         });
57         smUpdateStore.add(addDVDMenu);
58
59         menu.add(smUpdateStore);
60         JMenuItem viewStoreMenu= new JMenuItem("View store");
61         viewStoreMenu.addActionListener(e->{
62             c.showStoreScreen();
63         });
64         menu.add(viewStoreMenu);
65
66         JMenuItem viewCartMenu= new JMenuItem("View cart");
67         viewCartMenu.addActionListener(e->{
68             c.showCartScreen();
69         });
70         menu.add(viewCartMenu);
71
72         JMenuBar menuBar = new JMenuBar();
73         menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
74         menuBar.add(menu);
75
76         return menuBar;
77     }
78
79     JPanel createHeader() {
80         JPanel header = new JPanel();
81         header.setLayout(new BorderLayout());
82     }

```

AddDigitalVideoDiscToStoreScreen

```

1 package hust.soict.dsai.aims.screen;
2
3 import java.awt.Dimension;
4
15 public class AddDigitalVideoDiscToStoreScreen extends AddItemToStoreScreen {
16     private JTextField director;
17     private JTextField length;
18
19     public AddDigitalVideoDiscToStoreScreen(Store store, Cart cart, ControllerScreen c) {
20         super(store, cart, c);
21     }
22
23     @Override
24     void updateAdd(JPanel panel) {
25         this.numberInput = 6;
26         // SpringLayout layout = new SpringLayout();
27         // JPanel p = new JPanel(layout);
28         JLabel directorLabel = new JLabel("director", JLabel.TRAILING);
29         panel.add(directorLabel);
30         director = new JTextField(2);
31         director.setPreferredSize(new Dimension(150, 20));
32         directorLabel.setLabelFor(director);
33         panel.add(director);
34         JLabel lengthLabel = new JLabel("length", JLabel.TRAILING);
35         panel.add(lengthLabel);
36         length = new JTextField(2);
37         lengthLabel.setLabelFor(length);
38         panel.add(length);
39         JButton tes = new JButton("add");
40         tes.setVisible(false);
41         panel.add(tes);
42         panel.setPreferredSize(new Dimension(100, 300));
43         addBtn = new JButton("add");
44         addBtn.addActionListener(e -> {
45             addMedia();
46         });
47         panel.add(addBtn);
48     }
49
50     public void addMedia() {
51         String title = this.title.getText();
52         String director = this.director.getText();
53         String category = this.category.getText();
54         float cost = Float.parseFloat(this.cost.getText());
55         int length = Integer.parseInt(this.length.getText());
56         DigitalVideoDisc dvd = new DigitalVideoDisc(title, category, director, length, cost);
57         this.store.addMedia(dvd);
58         JOptionPane.showMessageDialog(null, "add DVD successfully!");
59         clearTextField();
60     }
61
62     public void clearTextField(){
63         this.title.setText("");
64         this.director.setText("");
65         this.cost.setText("");
66         this.length.setText("");
67         this.category.setText("");

```

AddCompactDiscToStoreScreen

```

1 package hust.soict.dsai.aims.screen;
2
3 import java.awt.Dimension;
4
15
16 public class AddCompactDiscToStoreScreen extends AddItemToStoreScreen {
17     private JTextField artist;
18     private JTextField listTrack;
19
20     public AddCompactDiscToStoreScreen(Store store, Cart cart, ControllerScreen c) {
21         super(store, cart, c);
22     }
23
24
25     @Override
26     void updateAdd(JPanel panel) {
27         this.numberInput = 6;
28
29         JLabel artistLabel = new JLabel("artist", JLabel.TRAILING);
30         panel.add(artistLabel);
31         artist = new JTextField(2);
32         artist.setPreferredSize(new Dimension(150, 20));
33         artistLabel.setLabelFor(artist);
34         panel.add(artist);
35         JLabel listTrackLabel = new JLabel("List track (each track separated by a comma Ex: track-length)",
36             JLabel.TRAILING);
37
38         panel.add(listTrackLabel);
39         listTrack = new JTextField(2);
40         listTrackLabel.setLabelFor(listTrack);
41         panel.add(listTrack);
42         JButton tes = new JButton("add");
43         tes.setVisible(false);
44         panel.add(tes);
45         panel.setPreferredSize(new Dimension(100, 300));
46         addBtn = new JButton("add");
47         addBtn.addActionListener(e -> {
48             addMedia();
49         });
50         panel.add(addBtn);
51     }
52
53     public void addMedia() {
54         String title = this.title.getText();
55         String listTrack = this.listTrack.getText();
56         String category = this.category.getText();
57         float cost = Float.parseFloat(this.cost.getText());
58         String artist = (this.artist.getText());
59         String[] arrayTrack = listTrack.trim().split(",");
60
61         CompactDisc cd = new CompactDisc(artist, title, category, "", cost);
62         ;
63         for (String track : arrayTrack) {
64             String titleTrack = track.split("-")[0].trim();
65             int lengthTrack = Integer.parseInt(track.split("-")[1].trim());
66             Track newTrack = new Track(titleTrack, lengthTrack);
67             cd.addTrack(newTrack);
68         }
69     }

```

AddBookToStoreScreen

```

1 package hust.soict.dsai.aims.screen;
2
3 import java.awt.Dimension;
4
15 public class AddBookToStoreScreen extends AddItemToStoreScreen {
16     private JTextField listAuthor;
17
18     public AddBookToStoreScreen(Store store, Cart cart, ControllerScreen c) {
19         super(store, cart, c);
20     }
21
22     @Override
23     void updateAdd(JPanel panel) {
24         this.numberInput = 5;
25
26         JLabel listAuthorLabel = new JLabel("Authors(Names are separated by a comma)", JLabel.TRAILING);
27         panel.add(listAuthorLabel);
28         listAuthor = new JTextField(2);
29         listAuthor.setPreferredSize(new Dimension(150, 20));
30         listAuthorLabel.setLabelFor(listAuthor);
31         panel.add(listAuthor);
32
33         JButton tes = new JButton("add");
34         tes.setVisible(false);
35         panel.add(tes);
36         panel.setPreferredSize(new Dimension(100, 300));
37         addBtn = new JButton("add");
38         addBtn.addActionListener(e -> {
39             addMediaToStore();
40         });
41         panel.add(addBtn);
42     }
43
44     public void addMediaToStore() {
45         String title = this.title.getText();
46         String listAuthor = this.listAuthor.getText();
47         String[] arrayAuthor = listAuthor.split(",");
48         String category = this.category.getText();
49         float cost = Float.parseFloat(this.cost.getText());
50         Book book = new Book(title, category, cost);
51         for(String author:arrayAuthor) {
52             book.addAuthor(author);
53         }
54         this.store.addMedia(book);
55         JOptionPane.showMessageDialog(null, "add Book successfully!");
56         clearTextField();
57     }
58
59     public void clearTextField(){
60         this.title.setText("");
61         this.listAuthor.setText("");
62         this.cost.setText("");
63         this.category.setText("");
64     }
65
66
67 }

```

12. Check all the previous source codes to catch/handle/delegate runtime exceptions

```

29
30 public void addMedia(Media media) throws LimitExceededException {
31     if ((itemsOrdered.size()) >= MAX_ORDERED) {
32         throw new LimitExceededException("Full");
33     }
34     else if (itemsOrdered.contains(media)) {
35         System.out.println("This is already in your order!");
36     }
37     else {
38         itemsOrdered.add(media);
39         System.out.println("Media added!");
40     }
41 }
42

```

13. Create a class which inherits from Exception

```
1 package hust.soict.dsai.aims.exception;
2
3
4 public class PlayerException extends Exception {
5
6     public PlayerException() {
7         // TODO Auto-generated constructor stub
8     }
9
10    public PlayerException(String message) {
11        super(message);
12        // TODO Auto-generated constructor stub
13    }
14
15    public PlayerException(Throwable cause) {
16        super(cause);
17        // TODO Auto-generated constructor stub
18    }
19
20    public PlayerException(String message, Throwable cause) {
21        super(message, cause);
22        // TODO Auto-generated constructor stub
23    }
24
25    public PlayerException(String message, Throwable cause, boolean enableSuppression, boolean writableStackTrace) {
26        super(message, cause, enableSuppression, writableStackTrace);
27        // TODO Auto-generated constructor stub
28    }
29
30 }
```

CompactDisc

```
1 @Override
2 public void play() throws PlayerException{
3     System.out.println("CompactDisc Artist: " + this.getArtist());
4     System.out.println("Total length: " + this.getLength());
5
6     if (this.getLength() > 0) {
7         System.out.println("Compactdisc: " + this.getTitle());
8         System.out.println("CompactDisc Artist: " + this.getArtist());
9         System.out.println("Total length: " + this.getLength());
10        java.util.Iterator iter = tracks.iterator();
11        Track nextTrack;
12        while (iter.hasNext()) {
13            nextTrack = (Track) iter.next();
14            try {
15                nextTrack.play();
16            }
17            catch (PlayerException e) {
18                throw e;
19            }
20        }
21    }
22    else {
23        throw new PlayerException("Error: DVD length is non-positive!");
24    }
25
26    System.out.println("-----Play All Tracks-----");
27    for (Track track: tracks) {
28        track.play();
29    }
30 }
```

DVD


```

3  @Override
4  public void play() throws PlayerException{
5      if (this.getLength() > 0) {
6          System.out.println("Playing DVD: " + this.getTitle());
7          System.out.println("DVD length: " + this.getLength());
8      }
9      else {throw new PlayerException("Error: DVD length is non-positive!");}
10 }

```

Track

```

5  @Override
6  public void play() throws PlayerException {
7      if (this.getLength() > 0) {
8          System.out.println("Playing track: " + this.getTitle());
9          System.out.println("Track length: " + this.getLength());
10     }
11     else {throw new PlayerException("Error: DVD length is non-positive!");}
12 }

```

14. Update the Aims class

```

switch (choice) {
    case 1:
        cart.addMedia(media);
        System.out.println("Media added to cart.");
        break;
    case 2:
        if (media instanceof Playable) {
            if(media instanceof DigitalVideoDisc){
                try {
                    ((DigitalVideoDisc) media).play();
                } catch (PlayerException e) {
                    // TODO Auto-generated catch block
                    e.getMessage();
                    e.toString();
                    e.printStackTrace();
                }
            }
            else if(media instanceof CompactDisc){
                try {
                    ((CompactDisc)media).play();
                } catch (PlayerException e) {
                    // TODO Auto-generated catch block
                    e.getMessage();
                    e.toString();
                    e.printStackTrace();
                }
            }
        } else {
            System.out.println("This media cannot be played.");
        }
        break;
    case 0:
        break;
    default:
        System.out.println("Invalid choice! Please choose a number between 0-2.");
}
} while (choice != 0);
}

```

```

public static void playMedia(Scanner scanner) {
    System.out.print("Enter the title of the media to play: ");
    String title = scanner.nextLine();
    Media media = store.searchByTitle(title);

    if (media != null) {
        if (media instanceof Playable) {
            if (media instanceof DigitalVideoDisc) {
                try {
                    ((DigitalVideoDisc) media).play();
                } catch (PlayerException e) {
                    // TODO Auto-generated catch block
                    e.getMessage();
                    e.toString();
                    e.printStackTrace();
                }
            }
            else if (media instanceof CompactDisc) {
                try {
                    ((CompactDisc) media).play();
                } catch (PlayerException e) {
                    // TODO Auto-generated catch block
                    e.getMessage();
                    e.toString();
                    e.printStackTrace();
                }
            }
            else {
                System.out.println("This media cannot be played.");
            }
        }
        else {
            System.out.println("Media not found in the store.");
        }
    }
}

```

15. Modify the equals() method of Media class

```

44
45 ● @Override
46 public boolean equals(Object o) {
47     if (o instanceof Media) {
48         Media media = (Media) o;
49         if (this.title.equals(media.title)) {
50             return true;
51         } else {
52             return false;
53         }
54     }
55     return false;
56 }
57

```

[illegible]