

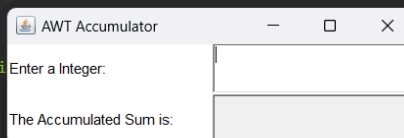
Lab05 Report

1. Swing components.

1.1. AWT Accumulator.

```
1 package hust.soict.dsai.swing;
2
3 import java.awt.Frame;
4
5 public class AWTAccumulator extends Frame {
6
7     private TextField tfInput;
8     private TextField tfOutput;
9     private int sum = 0;
10
11     public AWTAccumulator() {
12         setLayout(new GridLayout(2, 2));
13
14         add(new Label("Enter a Integer: "));
15
16         tfInput = new TextField(10);
17         add(tfInput);
18         tfInput.addActionListener(new TFInputListener());
19
20         add(new Label("The Accumulated Sum is: "));
21
22         tfOutput = new TextField(10);
23         tfOutput.setEditable(false);
24         add(tfOutput);
25
26         setTitle("AWT Accumulator");
27         setSize(350, 120);
28         setVisible(true);
29     }
30
31     public static void main(String[] args) {
32         new AWTAccumulator();
33     }
34
35     private class TFInputListener implements ActionListener {
36
37         @Override
38         public void actionPerformed(ActionEvent evt) {
39             int numberIn = Integer.parseInt(tfInput.getText());
40             sum += numberIn;
41             tfInput.setText("");
42             tfOutput.setText(sum + "");
43         }
44     }
45 }
```

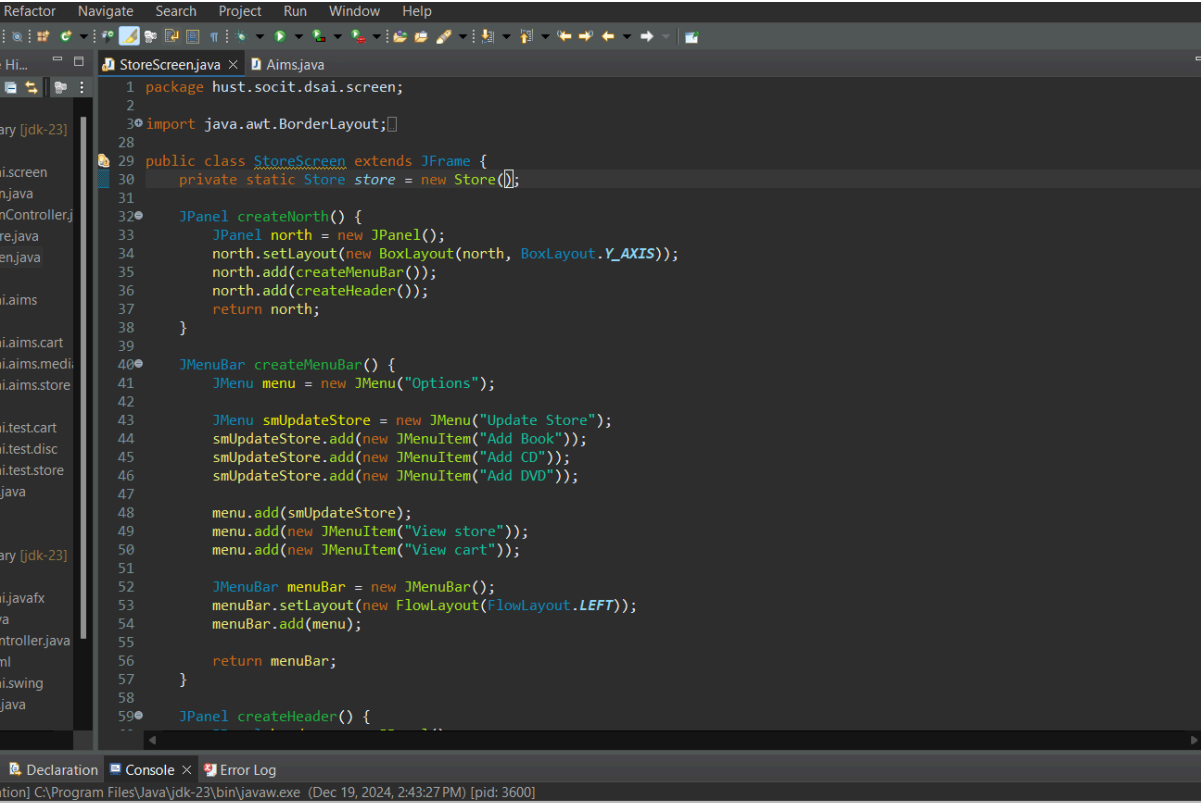
```
1 package hust.soict.dsai.swing;
2
3 import java.awt.Frame;
4
5 public class AWTAccumulator extends Frame {
6
7     private TextField tfInput;
8     private TextField tfOutput;
9     private int sum = 0;
10
11     public AWTAccumulator() {
12         setLayout(new GridLayout(2, 2));
13
14         add(new Label("Enter a Integer: "));
15
16         tfInput = new TextField(10);
17         add(tfInput);
18         tfInput.addActionListener(new TFInputListener());
19
20         add(new Label("The Accumulated Sum is: "));
21
22         tfOutput = new TextField(10);
23         tfOutput.setEditable(false);
24         add(tfOutput);
25
26         setTitle("AWT Accumulator");
27         setSize(350, 120);
28         setVisible(true);
29     }
30
31     private class TFInputListener implements ActionListener {
32
33         @Override
34         public void actionPerformed(ActionEvent evt) {
35             int numberIn = Integer.parseInt(tfInput.getText());
36             sum += numberIn;
37             tfInput.setText("");
38             tfOutput.setText(sum + "");
39         }
40     }
41 }
```



2. Create a graphical user interface for AIMS with Swing.

2.1. View Store Screen.

2.1.1. StoreScreen class.



```
1 package hust.socit.dsai.screen;
2
3 import java.awt.BorderLayout;
4
5 public class StoreScreen extends JFrame {
6     private static Store store = new Store();
7
8     JPanel createNorth() {
9         JPanel north = new JPanel();
10        north.setLayout(new BorderLayout(north, BorderLayout.Y_AXIS));
11        north.add(createMenuBar());
12        north.add(createHeader());
13        return north;
14    }
15
16    JMenuBar createMenuBar() {
17        JMenu menu = new JMenu("Options");
18
19        JMenu smUpdateStore = new JMenu("Update Store");
20        smUpdateStore.add(new JMenuItem("Add Book"));
21        smUpdateStore.add(new JMenuItem("Add CD"));
22        smUpdateStore.add(new JMenuItem("Add DVD"));
23
24        menu.add(smUpdateStore);
25        menu.add(new JMenuItem("View store"));
26        menu.add(new JMenuItem("View cart"));
27
28        JMenuBar menuBar = new JMenuBar();
29        menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
30        menuBar.add(menu);
31
32        return menuBar;
33    }
34
35    JPanel createHeader() {
```

Declaration Console Error Log

tion] C:\Program Files\Java\jdk-23\bin\javaw.exe (Dec 19, 2024, 2:43:27 PM) [pid: 3600]

```
StoreScreen.java x Aimsjava
58
59● JPanel createHeader() {
60     JPanel header = new JPanel();
61     header.setLayout(new BoxLayout(header, BoxLayout.X_AXIS));
62
63     JLabel title = new JLabel("AIMS");
64     title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 50));
65     title.setForeground(Color.CYAN);
66
67     JButton cart = new JButton("View cart");
68     cart.setPreferredSize(new Dimension(100, 50));
69     cart.setMaximumSize(new Dimension(100, 50));
70
71     header.add(Box.createRigidArea(new Dimension(10, 10)));
72     header.add(title);
73     header.add(Box.createHorizontalGlue());
74     header.add(cart);
75     header.add(Box.createRigidArea(new Dimension(10, 10)));
76
77     return header;
78 }
79
80● JPanel createCenter() {
81
82     JPanel center = new JPanel();
83     center.setLayout(new GridLayout(3, 3, 2, 2));
84
85
86     ArrayList<Media> mediaInStore = StoreScreen.store.getItemsInStore();
87     for (int i = 0; i < mediaInStore.size(); i++) {
88         MediaStore cell = new MediaStore(mediaInStore.get(i));
89         center.add(cell);
90     }
91
92     return center;
93 }
```

```
StoreScreen.java x Aimsjava
94
95● public static void initializeStore() {
96
97     DigitalVideoDisc dvd1 = new DigitalVideoDisc("The Lion King", "Animation", "Roger Allers", 87, 19.95f);
98     DigitalVideoDisc dvd2 = new DigitalVideoDisc("Star War", "Science Fiction", "George Lucas", 87, 24.95f);
99     DigitalVideoDisc dvd3 = new DigitalVideoDisc("Aladin", "Animation", 18.99f);
100     store.addMedia(dvd1);
101     store.addMedia(dvd2);
102     store.addMedia(dvd3);
103
104     CompactDisc cd1 = new CompactDisc("With you", "Music", "Twice", 15.26f);
105     Track track1Cd1 = new Track("One Spark", 3);
106     Track track2Cd1 = new Track("I Got You", 3);
107     Track track3Cd1 = new Track("Bloom", 2);
108     Track track4Cd1 = new Track("Rush", 3);
109     cd1.addTrack(track1Cd1);
110     cd1.addTrack(track2Cd1);
111     cd1.addTrack(track3Cd1);
112     cd1.addTrack(track4Cd1);
113
114     CompactDisc cd2 = new CompactDisc("Say Yes", "Music", "Twice", 14.67f);
115     Track track1Cd2 = new Track("Yes or Yes", 3);
116     Track track2Cd2 = new Track("Lalala", 3);
117     Track track3Cd2 = new Track("Say You Love Me", 2);
118     cd2.addTrack(track1Cd2);
119     cd2.addTrack(track2Cd2);
120     cd2.addTrack(track3Cd2);
121
122     CompactDisc cd3 = new CompactDisc("Eyes Wide Open", "Music", "Twice", 16.67f);
123     Track track1Cd3 = new Track("I can't stop me", 3);
124     Track track2Cd3 = new Track("Cry For Me", 3);
125     Track track3Cd3 = new Track("Go Hard", 2);
126     cd3.addTrack(track1Cd3);
127     cd3.addTrack(track2Cd3);
128     cd3.addTrack(track3Cd3);
129 }
```

```

128         cd3.addTrack(track3Cd3);
129
130         store.addMedia(cd1);
131         store.addMedia(cd2);
132         store.addMedia(cd3);
133
134         Book book1 = new Book("Harry Potter: The Prequel", "Fantasy Novels", 20.34f);
135         Book book2 = new Book("Harry Potter and the Sorcerer's Stone", "Fantasy Novels", 21.44f);
136         Book book3 = new Book("Christmas at Hogwarts", "Fantasy Novels", 18.42f);
137         store.addMedia(book1);
138         store.addMedia(book2);
139         store.addMedia(book3);
140     }
141
142     public StoreScreen(Store store) {
143         StoreScreen.store = store;
144
145         Container cp = getContentPane();
146         cp.setLayout(new BorderLayout());
147         cp.setLayout(new BorderLayout());
148
149         cp.add(createNorth(), BorderLayout.NORTH);
150         cp.add(createCenter(), BorderLayout.CENTER);
151
152         setVisible(true);
153         setTitle("Store");
154         setSize(1024, 768);
155     }
156
157     public static void main(String[] args) {
158         initializeStore();
159         new StoreScreen(store);
160     }
161 }
162

```

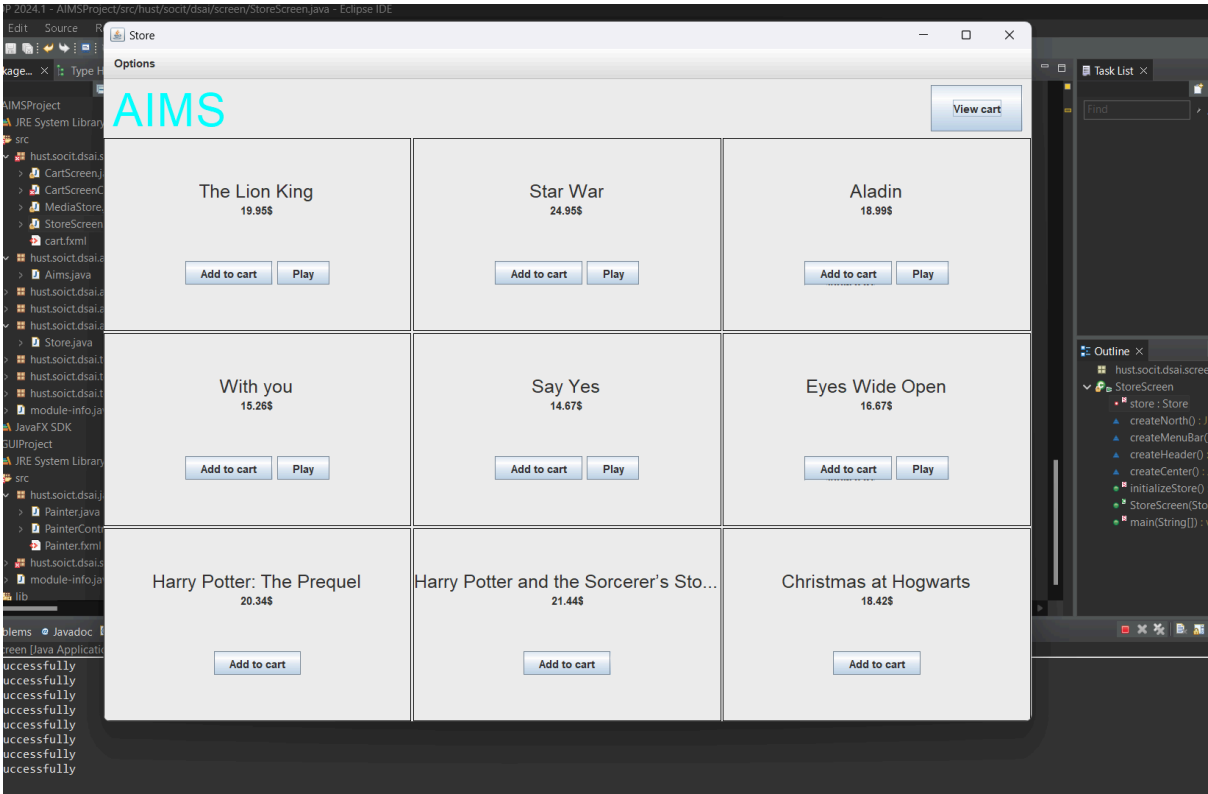
2.1.2. MediaStore class.

```

1 package hust.socit.dsai.screen;
2
3 import java.awt.Color;
4
5
6
7
8
9
10
11
12
13
14
15
16
17 public class MediaStore extends JPanel{
18     private Media media;
19     public MediaStore(Media media) {
20         this.media = media;
21         this.setLayout(new BoxLayout(this, BoxLayout.Y_AXIS));
22
23         JLabel title = new JLabel(media.getTitle());
24         title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 20));
25         title.setAlignmentX(CENTER_ALIGNMENT);
26
27         JLabel cost = new JLabel(""+media.getCost()+"$");
28         cost.setAlignmentX(CENTER_ALIGNMENT);
29
30         JPanel container = new JPanel();
31         container.setLayout(new FlowLayout(FlowLayout.CENTER));
32
33         container.add(new JButton("Add to cart"));
34         if(media instanceof Playable) {
35             container.add(new JButton("Play"));
36         }
37
38         this.add(Box.createVerticalGLue());
39         this.add(title);
40         this.add(cost);
41         this.add(Box.createVerticalGLue());
42         this.add(container);
43
44         this.setBorder(BorderFactory.createLineBorder(Color.BLACK));
45     }
46 }
47

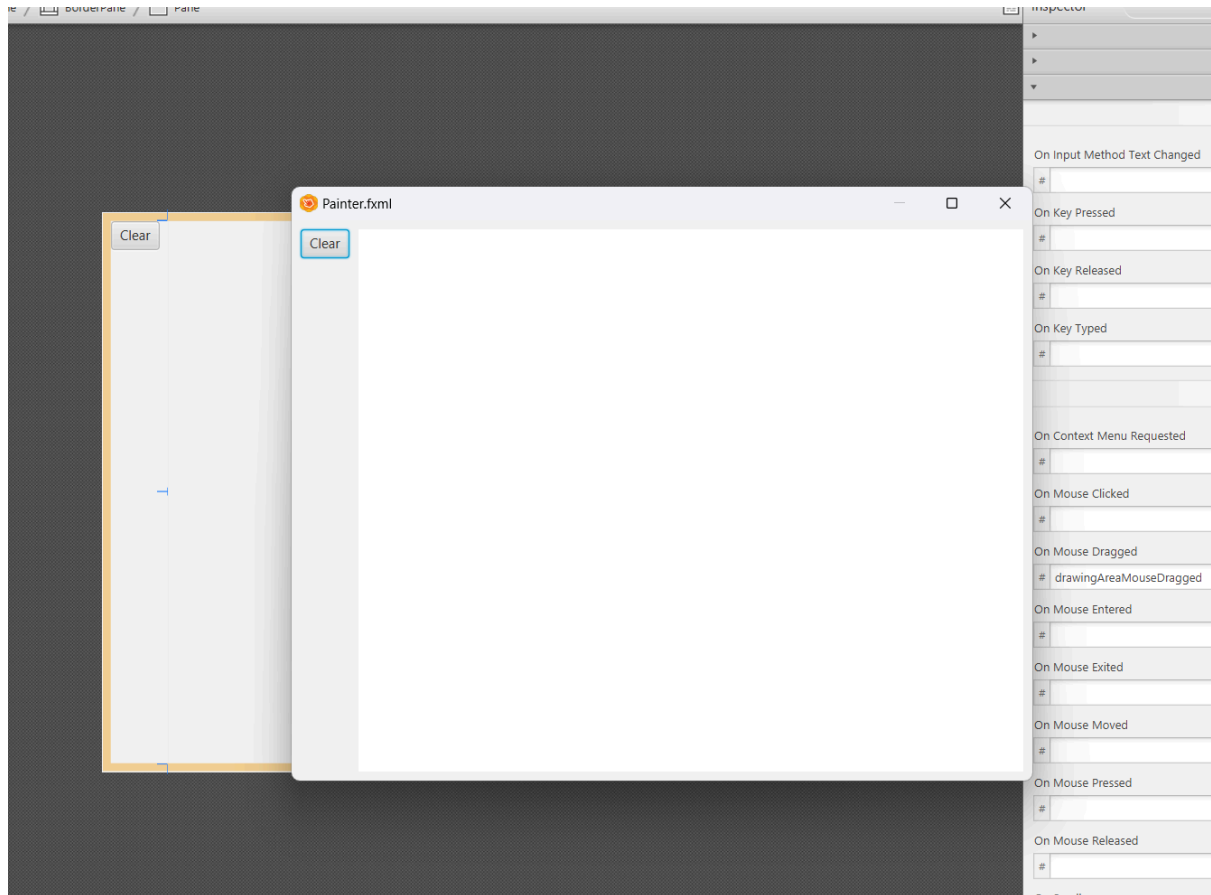
```

2.1.3. Store screen.

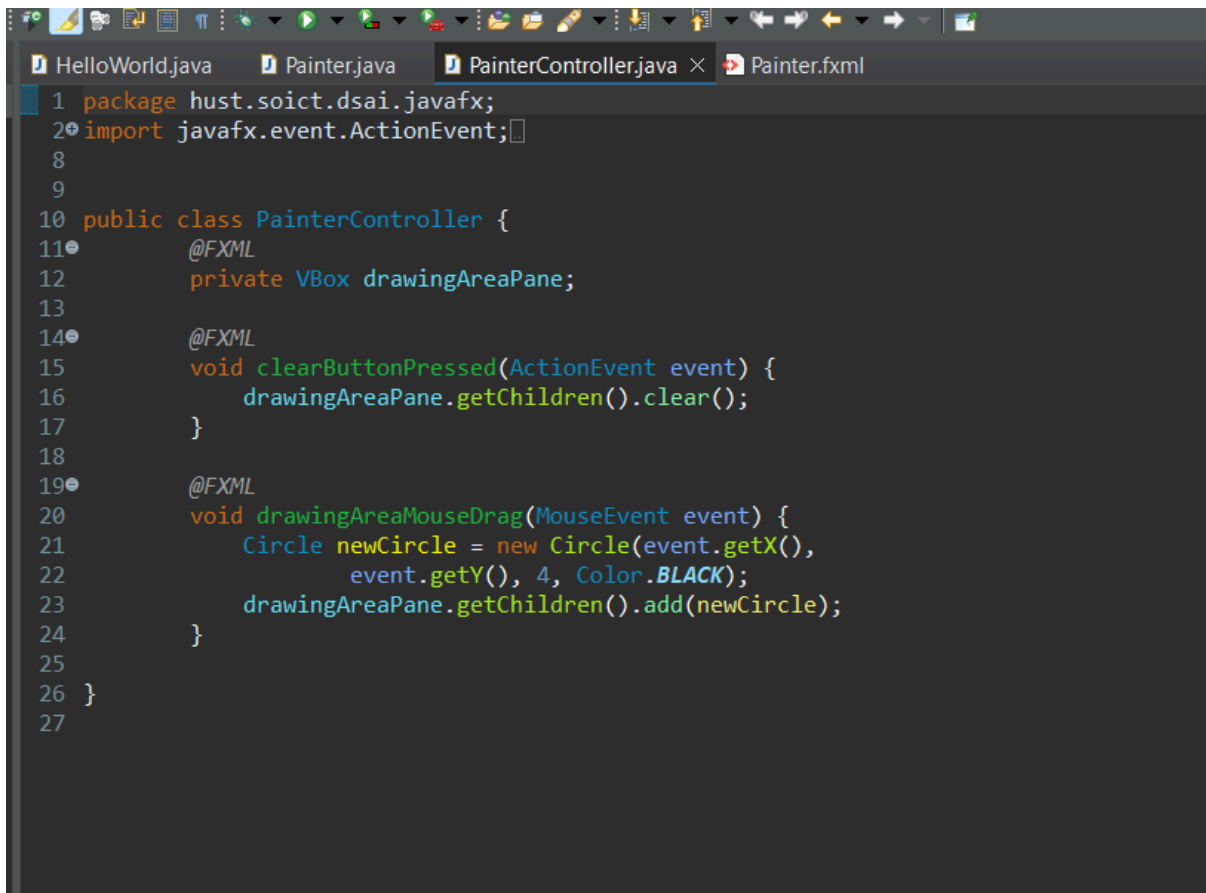


3. JavaFX API.

3.1. Create FXML file.

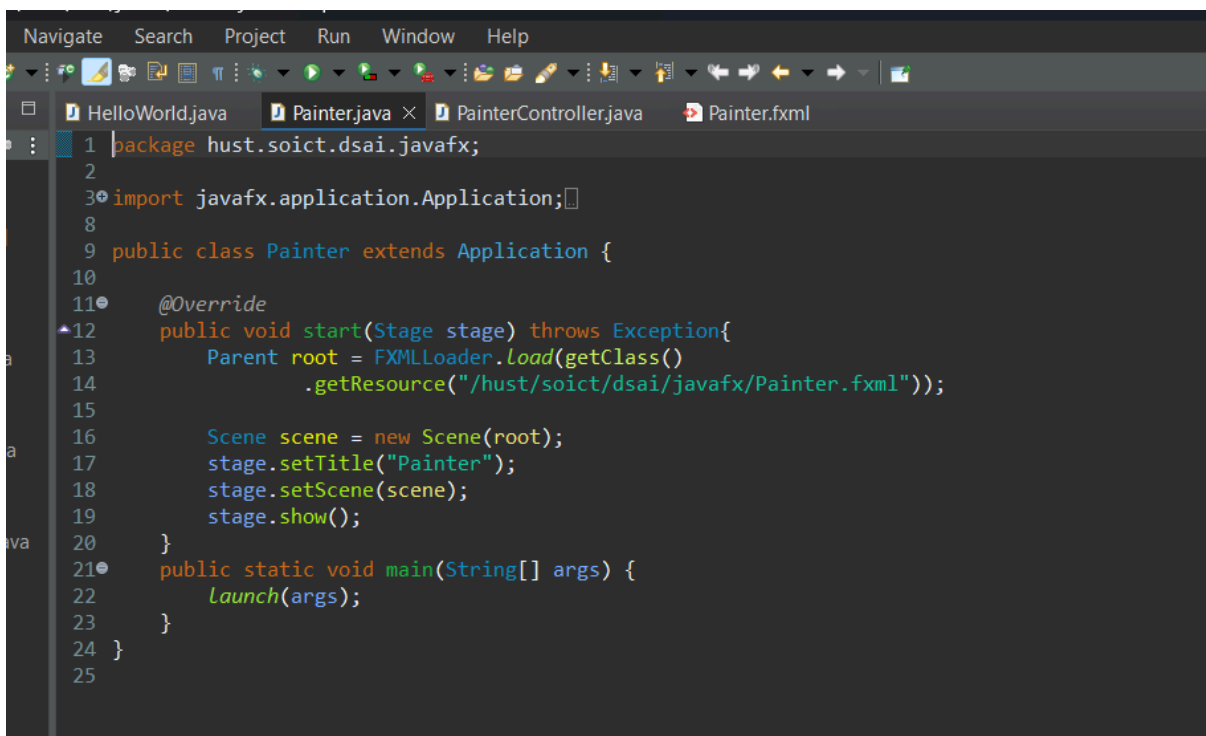


3.2. Create the controller class.

A screenshot of an IDE window showing the PainterController.java file. The file is part of the package hust.soict.dsai.javafx. It imports javafx.event.ActionEvent. The class PainterController has two methods annotated with @FXML: clearButtonPressed, which calls drawingAreaPane.getChildren().clear(), and drawingAreaMouseDrag, which creates a new black circle and adds it to drawingAreaPane.getChildren(). The drawingAreaPane is a private VBox field.

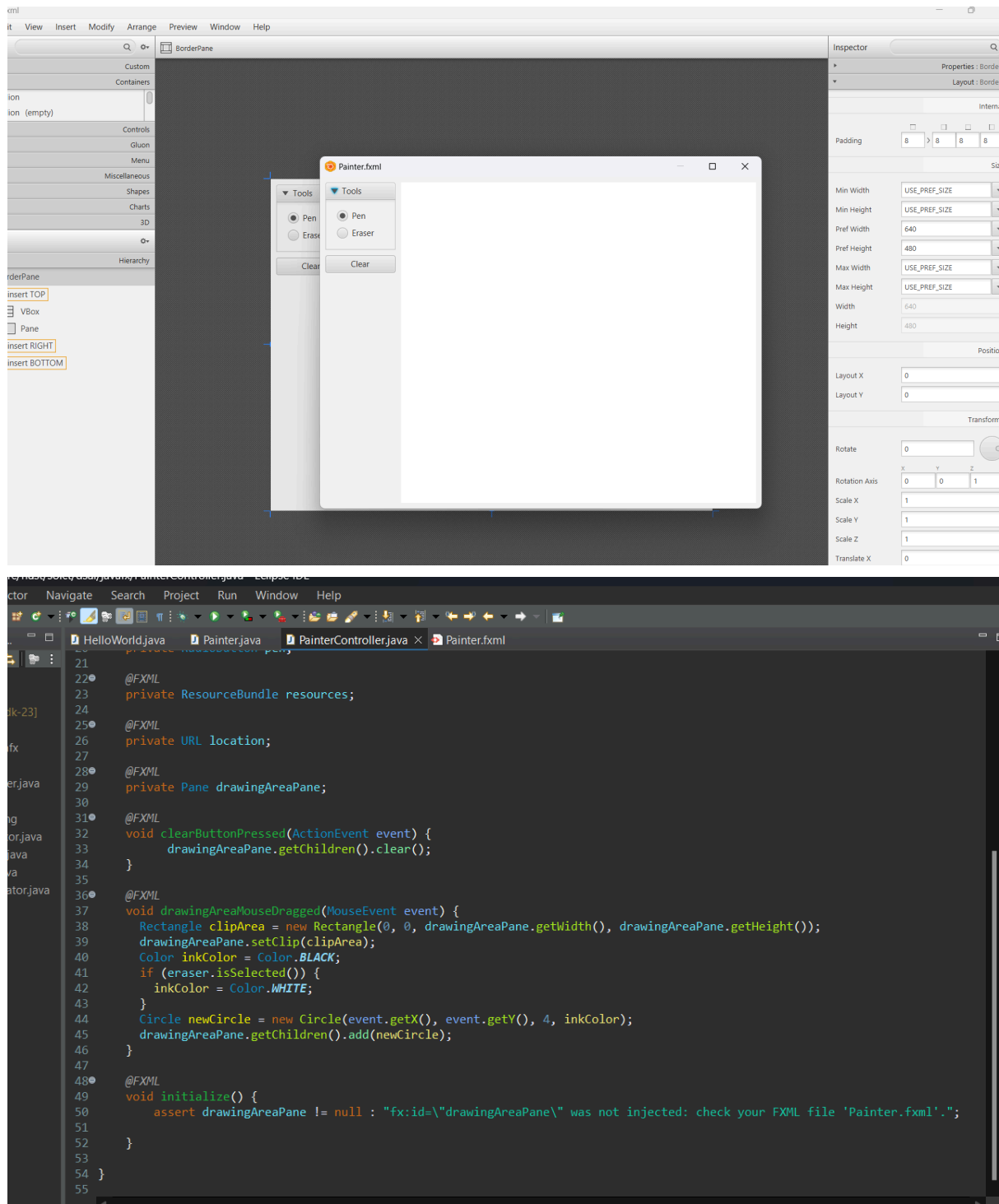
```
1 package hust.soict.dsai.javafx;
2 import javafx.event.ActionEvent;
3
4
5
6
7
8
9
10 public class PainterController {
11     @FXML
12     private VBox drawingAreaPane;
13
14     @FXML
15     void clearButtonPressed(ActionEvent event) {
16         drawingAreaPane.getChildren().clear();
17     }
18
19     @FXML
20     void drawingAreaMouseDrag(MouseEvent event) {
21         Circle newCircle = new Circle(event.getX(),
22             event.getY(), 4, Color.BLACK);
23         drawingAreaPane.getChildren().add(newCircle);
24     }
25 }
26
27
```

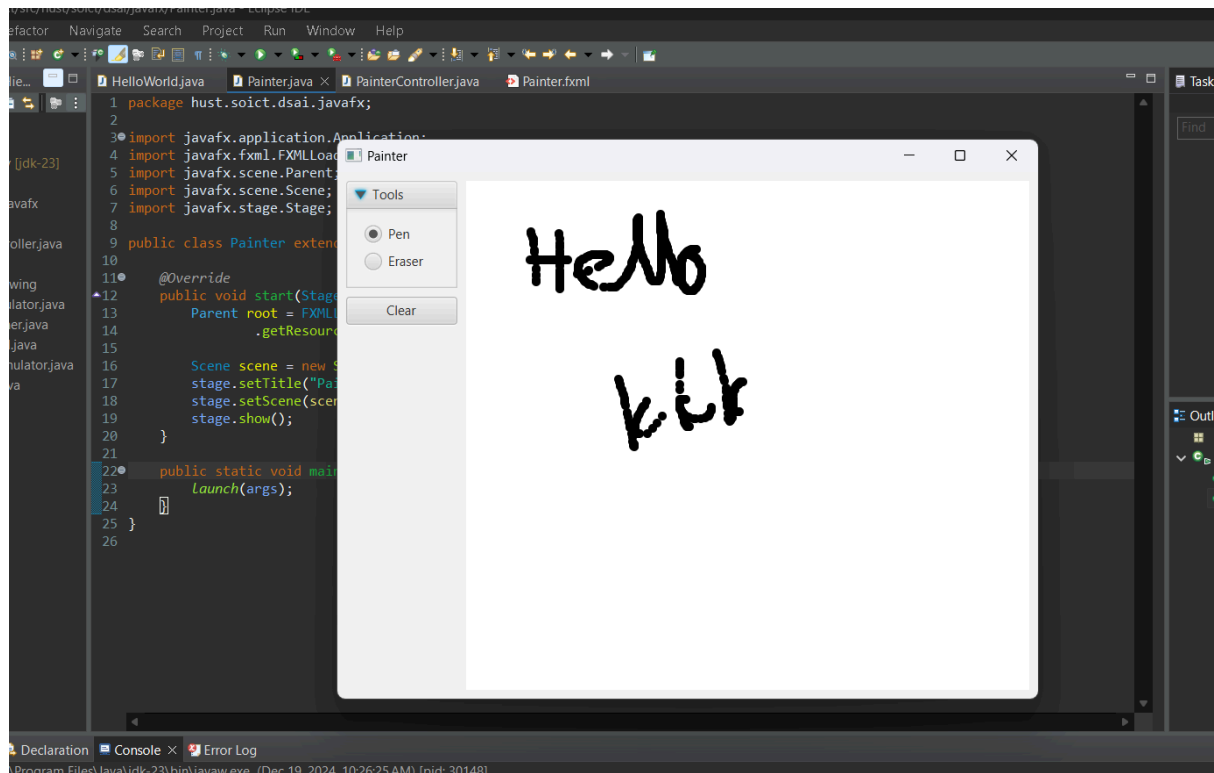
3.3. Create the application.

A screenshot of an IDE window showing the Painter.java file. The file is part of the package hust.soict.dsai.javafx. It imports javafx.application.Application. The class Painter extends Application and has two methods: start, which loads the Painter.fxml resource, creates a Scene, sets the stage title to "Painter", sets the scene, and shows the stage; and main, which calls launch. The start method is annotated with @Override.

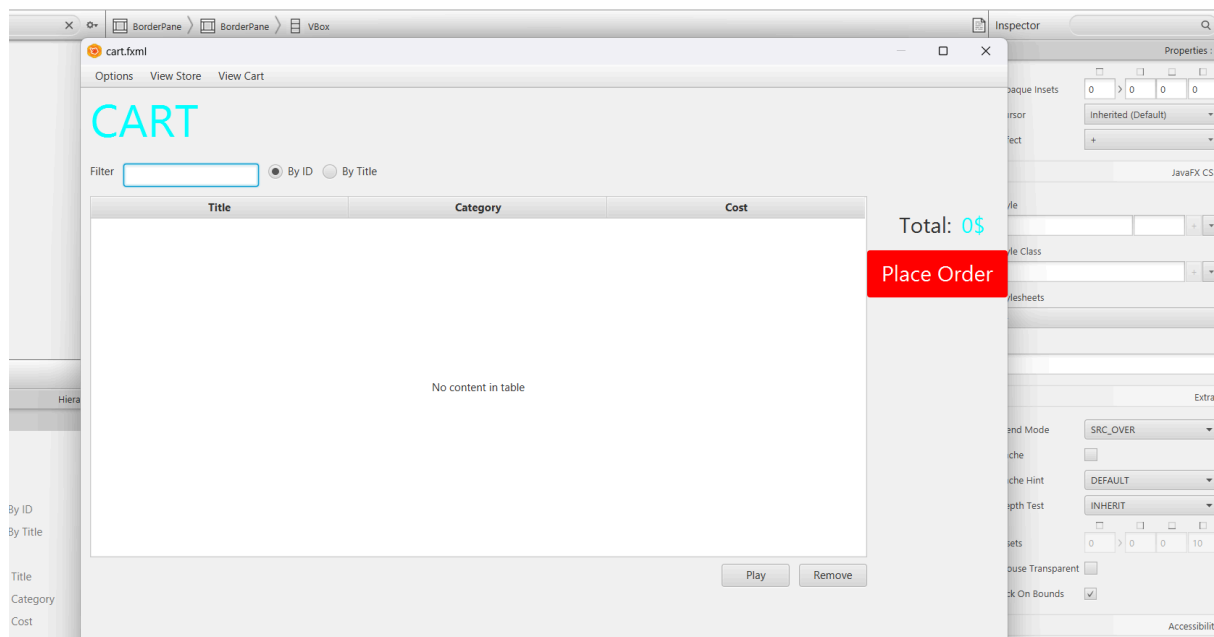
```
1 package hust.soict.dsai.javafx;
2
3 import javafx.application.Application;
4
5
6
7
8
9 public class Painter extends Application {
10
11     @Override
12     public void start(Stage stage) throws Exception {
13         Parent root = FXMLLoader.load(getClass()
14             .getResource("/hust/soict/dsai/javafx/Painter.fxml"));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

3.4. Practice ex.

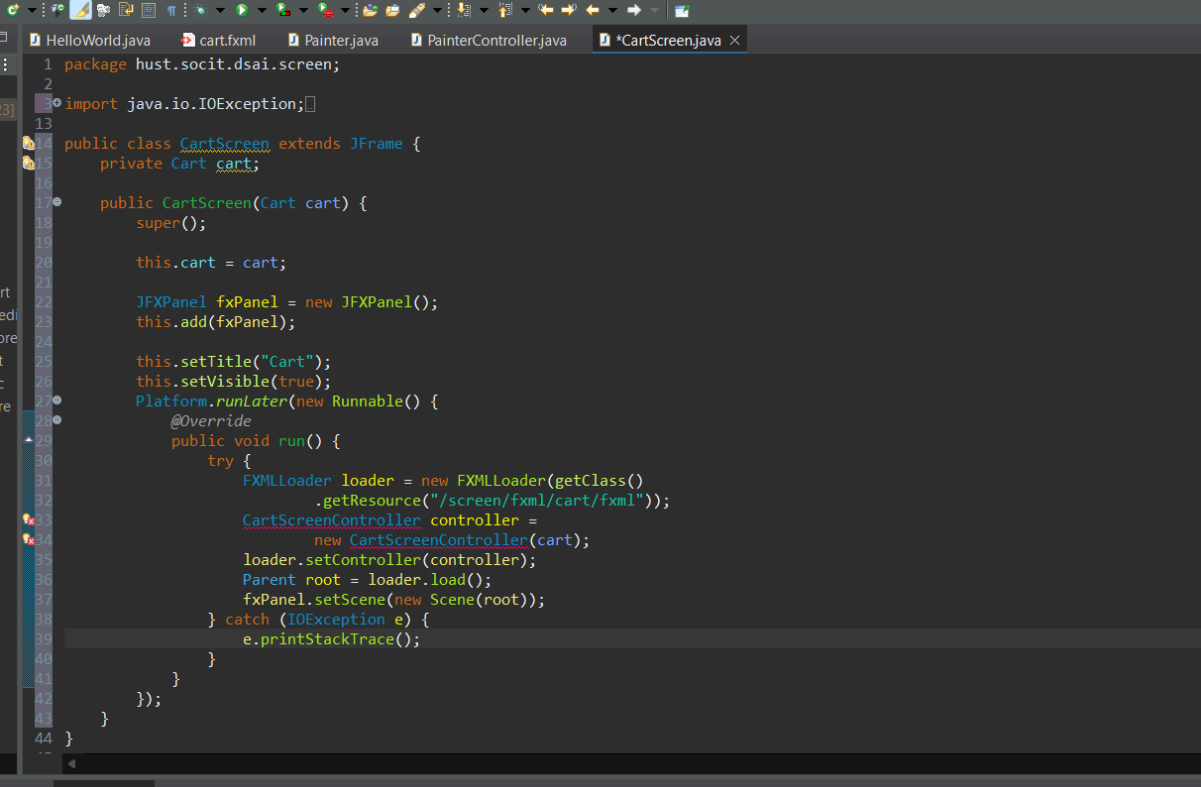




4. Setting up the View Cart Screen with ScreenBuilder.

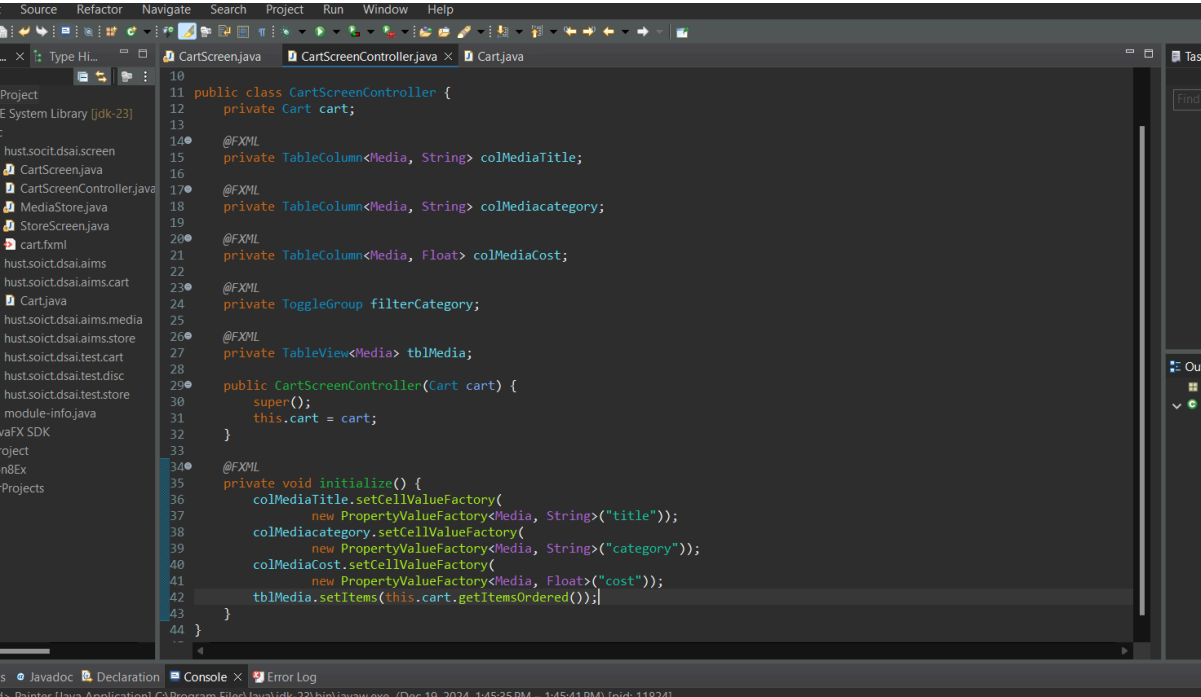


5. Integrating JavaFX into Swing application – The JFXPanel class.



```
1 package hust.socit.dsai.screen;
2
3 import java.io.IOException;
4
5 public class CartScreen extends JFrame {
6     private Cart cart;
7
8     public CartScreen(Cart cart) {
9         super();
10
11         this.cart = cart;
12
13         JFXPanel fxPanel = new JFXPanel();
14         this.add(fxPanel);
15
16         this.setTitle("Cart");
17         this.setVisible(true);
18         Platform.runLater(new Runnable() {
19             @Override
20             public void run() {
21                 try {
22                     FXMLLoader loader = new FXMLLoader(getClass()
23                         .getResource("/screen/fxml/cart/fxml"));
24                     CartScreenController controller =
25                         new CartScreenController(cart);
26                     loader.setController(controller);
27                     Parent root = loader.load();
28                     fxPanel.setScene(new Scene(root));
29                 } catch (IOException e) {
30                     e.printStackTrace();
31                 }
32             }
33         });
34     }
35 }
36
37
38
39
40
41
42
43
44 }
```

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



```
10 public class CartScreenController {
11     private Cart cart;
12
13     @FXML
14     private TableColumn<Media, String> colMediaTitle;
15
16     @FXML
17     private TableColumn<Media, String> colMediacategory;
18
19     @FXML
20     private TableColumn<Media, Float> colMediaCost;
21
22     @FXML
23     private ToggleGroup filterCategory;
24
25     @FXML
26     private TableView<Media> tblMedia;
27
28     public CartScreenController(Cart cart) {
29         super();
30         this.cart = cart;
31     }
32
33     @FXML
34     private void initialize() {
35         colMediaTitle.setCellValueFactory(
36             new PropertyValueFactory<Media, String>("title"));
37         colMediacategory.setCellValueFactory(
38             new PropertyValueFactory<Media, String>("category"));
39         colMediaCost.setCellValueFactory(
40             new PropertyValueFactory<Media, Float>("cost"));
41         tblMedia.setItems(this.cart.getItemsOrdered());
42     }
43 }
44
45
46
47
48
49
50
51
52 }
```

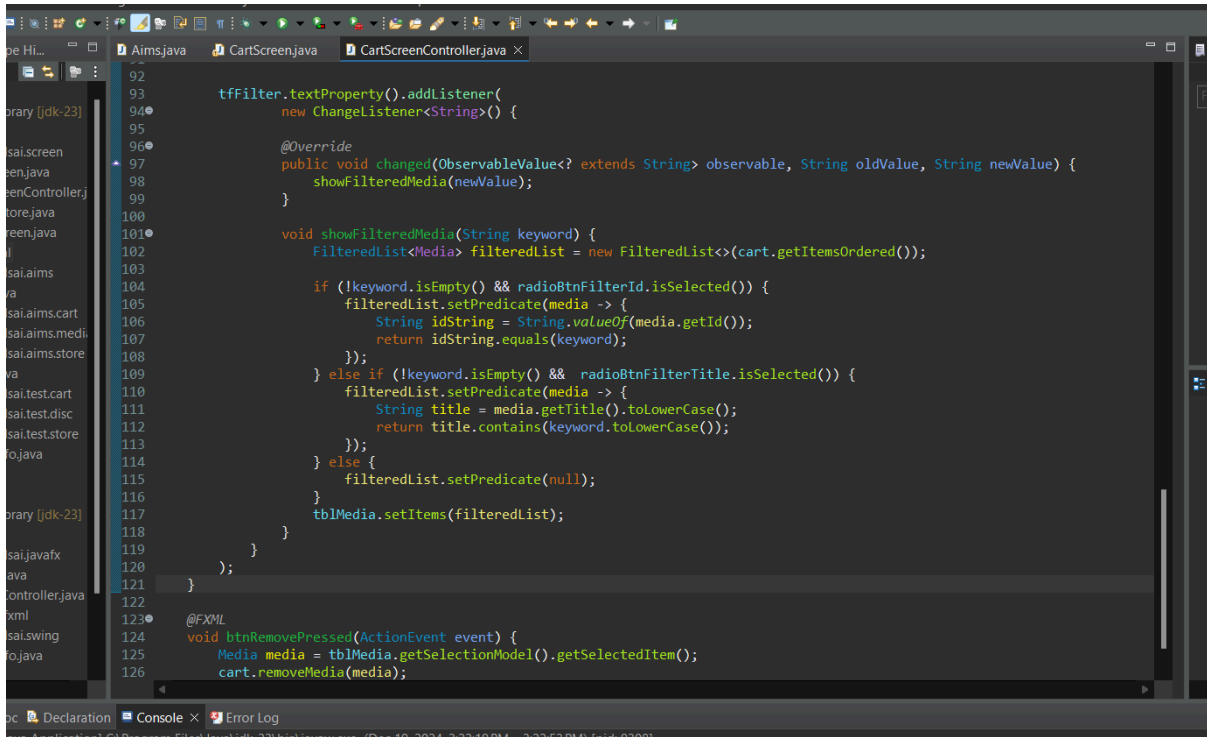
7. Updating buttons based on selected item in **TableView – ChangeListener**.

```
46 @FXML
47 private void initialize() {
48     colMediaTitle.setCellValueFactory(
49         new PropertyValueFactory<Media, String>("title"));
50     colMediaCategory.setCellValueFactory(
51         new PropertyValueFactory<Media, String>("category"));
52     colMediaCost.setCellValueFactory(
53         new PropertyValueFactory<Media, Float>("cost"));
54     tblMedia.setItems(this.cart.getItemsOrdered());
55
56     btnPlay.setVisible(false);
57     btnRemove.setVisible(false);
58
59     tblMedia.getSelectionModel().selectedItemProperty().addListener(
60         new ChangeListener<Media>() {
61
62             @Override
63             public void changed(ObservableValue<? extends Media> observable, Media oldValue, Media newValue) {
64                 if (newValue != null) {
65                     updateButtonBar(newValue);
66                 }
67             }
68
69             void updateButtonBar(Media media) {
70                 btnRemove.setVisible(true);
71                 if (media instanceof Playable) {
72                     btnPlay.setVisible(true);
73                 } else {
74                     btnPlay.setVisible(false);
75                 }
76             }
77         }
78     );
79 }
80
```

8. Deleting a Media.

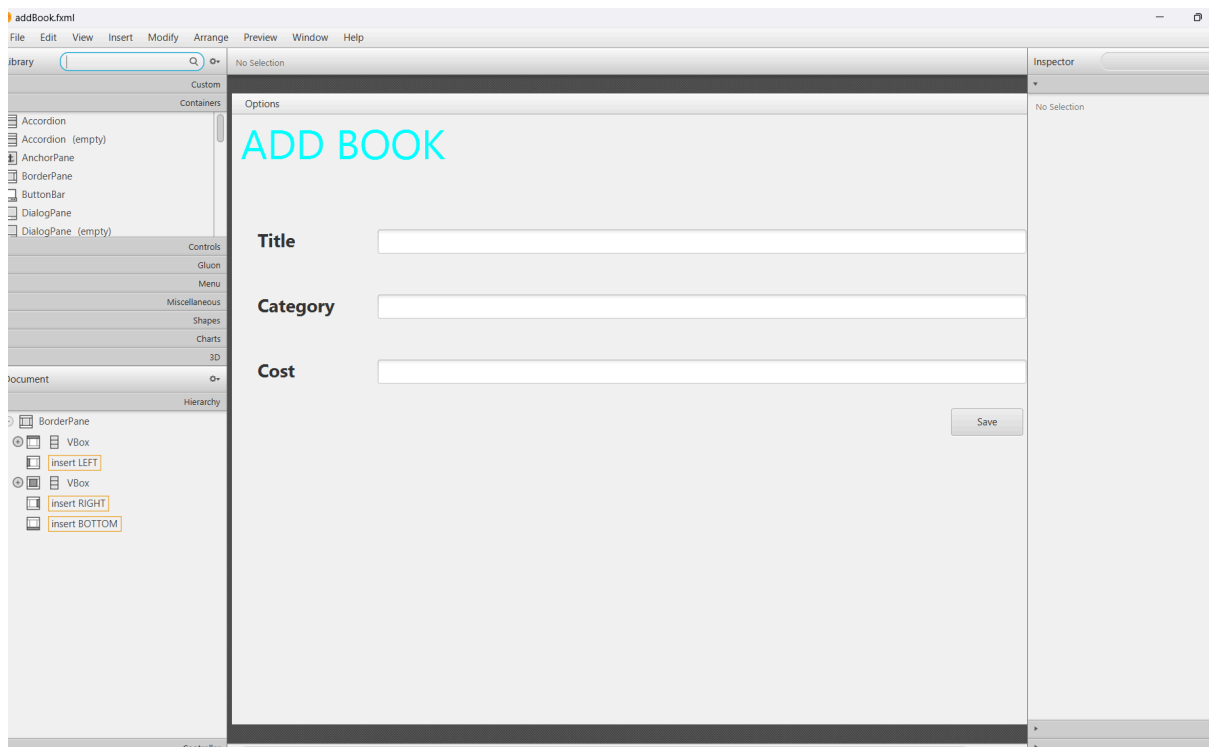
```
@FXML
void btnRemovePressed(ActionEvent event) {
    Media media = tblMedia.getSelectionModel().getSelectedItem();
    cart.removeMedia(media);
}
```

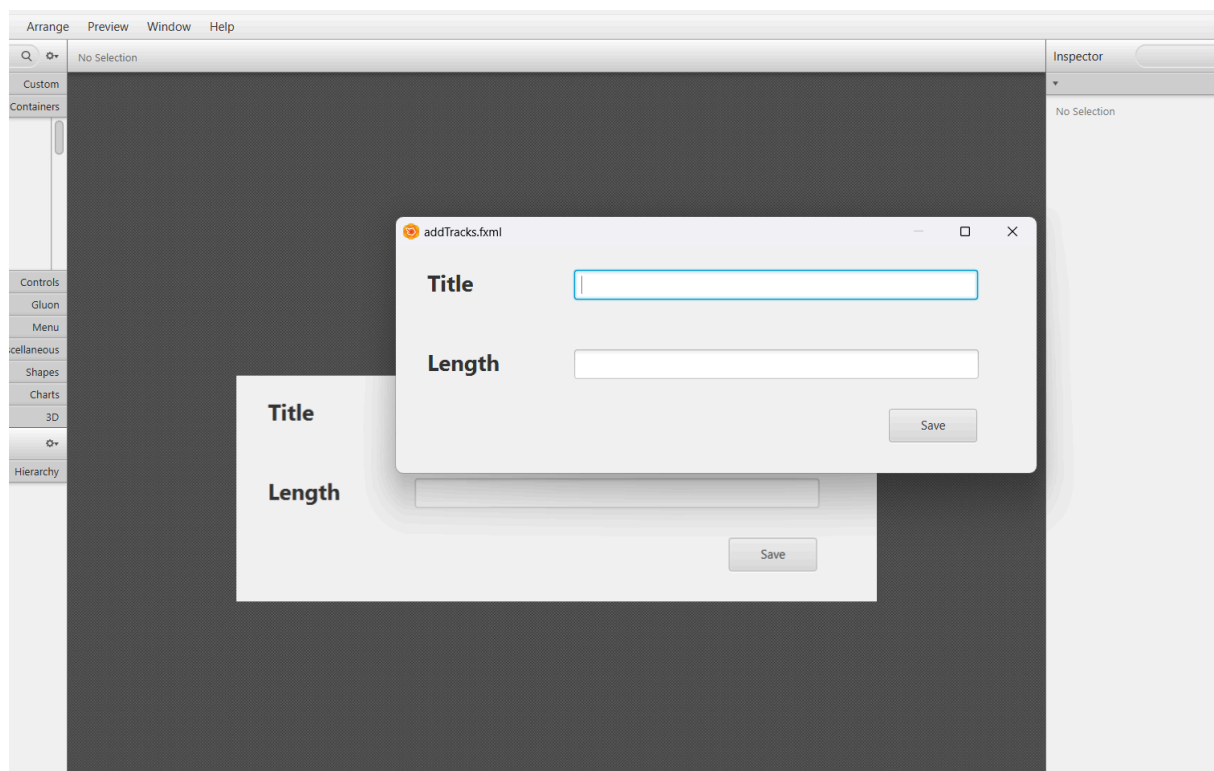
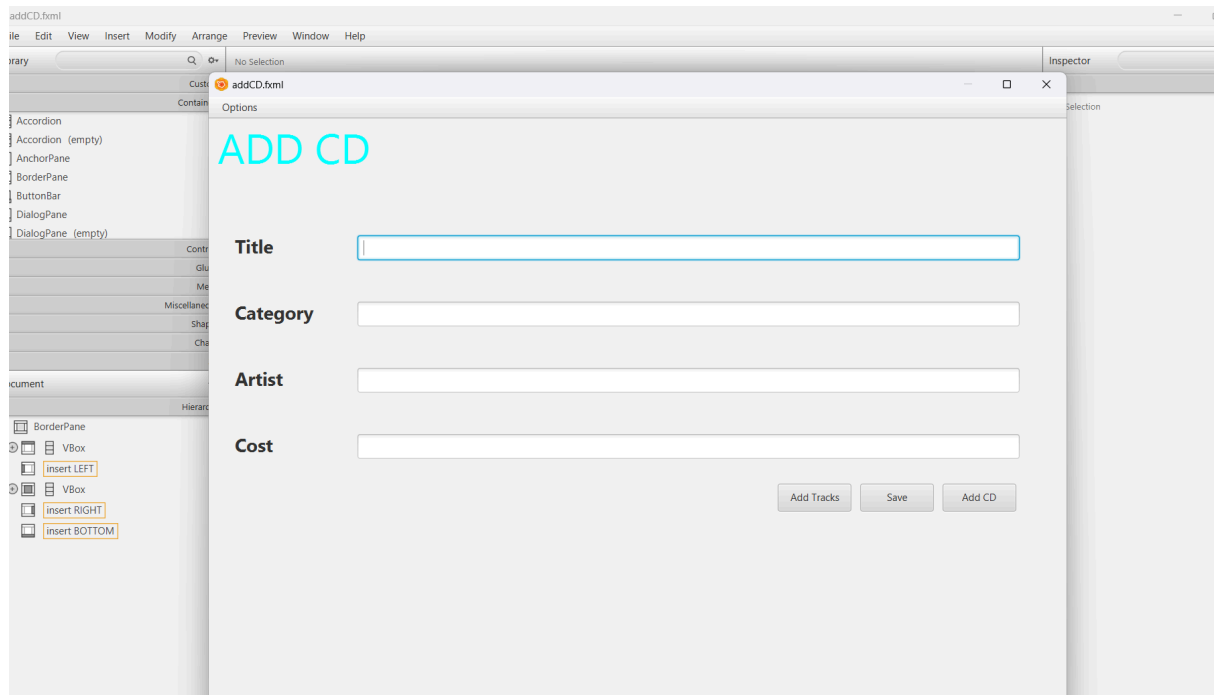
9. Filter items in cart – **FilteredList**.

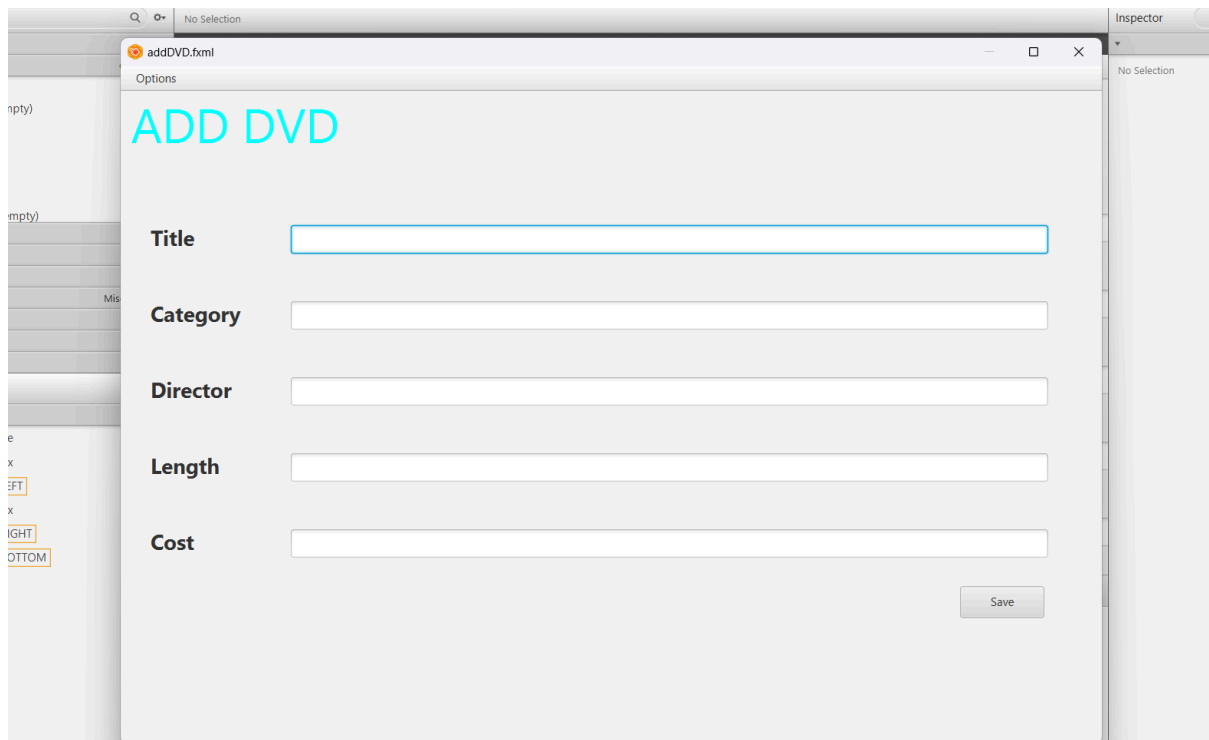


```
92  
93  
94     tffilter.textProperty().addListener(  
95         new ChangeListener<String>() {  
96  
97             @Override  
98             public void changed(ObservableValue<? extends String> observable, String oldValue, String newValue) {  
99                 showFilteredMedia(newValue);  
100             }  
101  
102             void showFilteredMedia(String keyword) {  
103                 FilteredList<Media> filteredList = new FilteredList<>(cart.getItemsOrdered());  
104  
105                 if (!keyword.isEmpty() && radioBtnFilterId.isSelected()) {  
106                     filteredList.setPredicate(media -> {  
107                         String idString = String.valueOf(media.getId());  
108                         return idString.equals(keyword);  
109                     });  
110                 } else if (!keyword.isEmpty() && radioBtnFilterTitle.isSelected()) {  
111                     filteredList.setPredicate(media -> {  
112                         String title = media.getTitle().toLowerCase();  
113                         return title.contains(keyword.toLowerCase());  
114                     });  
115                 } else {  
116                     filteredList.setPredicate(null);  
117                 }  
118                 tblMedia.setItems(filteredList);  
119             }  
120         });  
121     }  
122  
123     @FXML  
124     void btnRemovePressed(ActionEvent event) {  
125         Media media = tblMedia.getSelectionModel().getSelectedItem();  
126         cart.removeMedia(media);  
127     }  
128 }
```

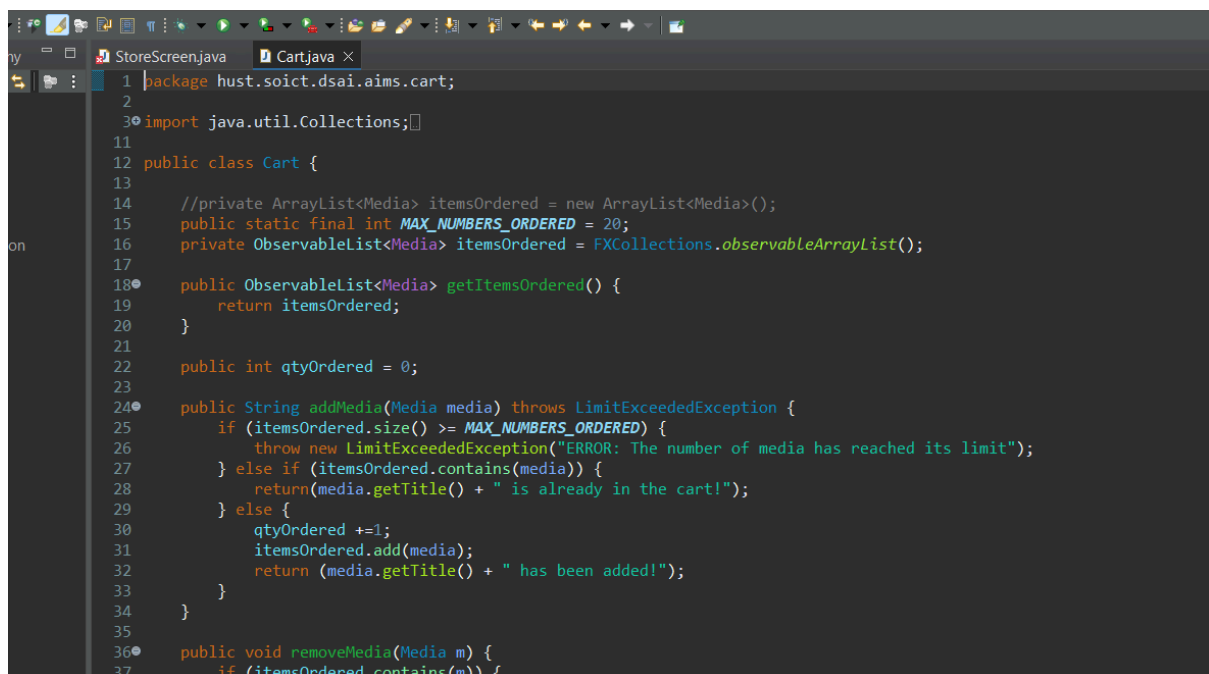
10. Complete the Aims GUI application.



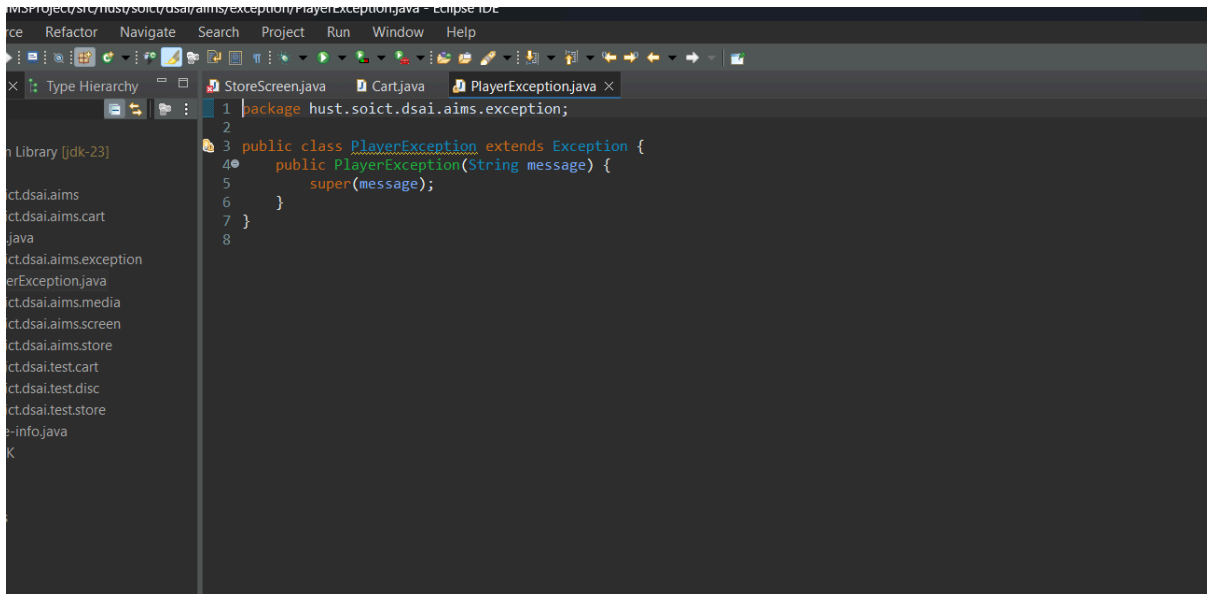




11. Check all the previous source codes to catch/handle/delegate runtime exceptions.

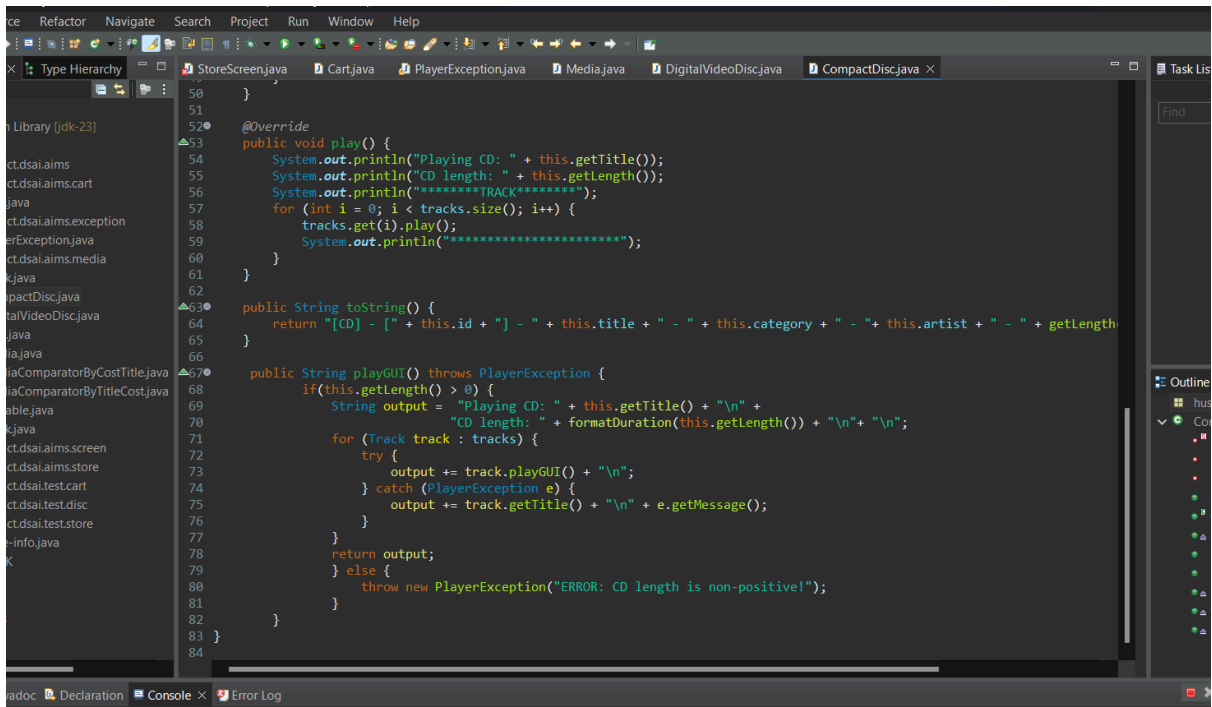


12. Create a class which inherits from **Exception**.



This screenshot shows the Eclipse IDE with the file `PlayerException.java` open. The code defines a class `PlayerException` that inherits from `Exception`. The class has a constructor `PlayerException(String message)` that calls `super(message)`. The package is `hust.soict.dsai.aims.exception`.

```
1 package hust.soict.dsai.aims.exception;
2
3 public class PlayerException extends Exception {
4     public PlayerException(String message) {
5         super(message);
6     }
7 }
8
```



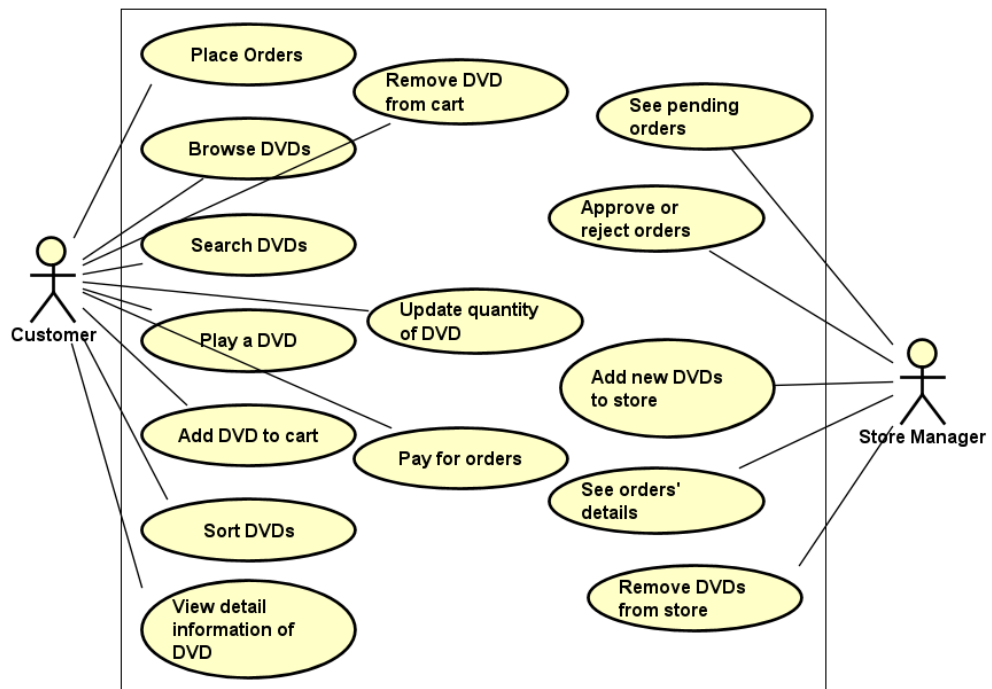
This screenshot shows the Eclipse IDE with the file `CompactDisc.java` open. The code implements the `CompactDisc` class, which inherits from `Media`. It includes methods `play()`, `toString()`, and `playGUI()`. The `play()` method prints the CD title, length, and tracks. The `toString()` method returns a string representation of the CD. The `playGUI()` method prints the CD title, length, and tracks, and throws a `PlayerException` if the CD length is non-positive.

```
50 }
51
52 @Override
53 public void play() {
54     System.out.println("Playing CD: " + this.getTitle());
55     System.out.println("CD length: " + this.getLength());
56     System.out.println("*****TRACK*****");
57     for (int i = 0; i < tracks.size(); i++) {
58         tracks.get(i).play();
59     }
60     System.out.println("*****");
61 }
62
63 public String toString() {
64     return "[CD] - [" + this.id + "] - " + this.title + " - " + this.category + " - " + this.artist + " - " + getLength();
65 }
66
67 public String playGUI() throws PlayerException {
68     if (this.getLength() > 0) {
69         String output = "Playing CD: " + this.getTitle() + "\n" +
70             "CD length: " + formatDuration(this.getLength()) + "\n" + "\n";
71         for (Track track : tracks) {
72             try {
73                 output += track.playGUI() + "\n";
74             } catch (PlayerException e) {
75                 output += track.getTitle() + "\n" + e.getMessage();
76             }
77         }
78         return output;
79     } else {
80         throw new PlayerException("ERROR: CD length is non-positive!");
81     }
82 }
83 }
84
```

```
18      this.title = title;
19      this.length = length;
20  }
21
22  @Override
23  public void play() {
24      System.out.println("Playing Track: " + this.getTitle());
25      System.out.println("Track's length: " + this.getLength());
26  }
27
28  public String formatDuration(int durationInSeconds) {
29      Duration duration = Duration.ofSeconds(durationInSeconds);
30      return String.format("%02d:%02d", duration.toMinutes(), duration.minusMinutes(duration.toMinutes()).getSeconds());
31  }
32
33  public boolean equals(Object obj) {
34      if (this == obj) {
35          return true;
36      }
37      if (!(obj instanceof Track)) {
38          return false;
39      }
40      return ((Track)obj).getTitle() == this.getTitle() && ((Track)obj).getLength() == this.getLength();
41  }
42
43  public String playGUI() throws PlayerException {
44      if (this.getLength() > 0) {
45          return "Playing track: " + this.getTitle() + "\n" +
46              "Track length: " + formatDuration(this.getLength());
47      } else {
48          throw new PlayerException("ERROR: Track length is non-positive!");
49      }
50  }
51  }
52  }
```

```
28      this.category = category;
29      this.director = director;
30      this.cost = cost;
31  }
32  public DigitalVideoDisc(String title, String category, String director, int length, float cost) {
33      super(title, category, director, length, cost);
34      this.id = nbDigitalVideoDiscs;
35      nbDigitalVideoDiscs++;
36      this.title = title;
37      this.category = category;
38      this.director = director;
39      this.length = length;
40      this.cost = cost;
41  }
42  @Override
43  public void play() {
44      System.out.println("Playing DVD: " + this.getTitle());
45      System.out.println("DVD length: " + this.getLength());
46  }
47
48  public String toString() {
49      return "[DVD] - [" + this.id + "] - " + this.title + " - " + this.category + " - " + this.director
50          + " - " + this.length + ": " + this.cost + "$";
51  }
52
53  public String playGUI() throws PlayerException {
54      if (this.getLength() > 0) {
55          return "Playing DVD: " + this.getTitle() + "\n" +
56              "DVD length: " + formatDuration(this.getLength());
57      } else {
58          throw new PlayerException("ERROR: DVD length is non-positive!");
59      }
60  }
61  }
62  }
```


uc



pkg

