Report Lab05

Student's name: Le Viet Cuong
Student's id: 20235586

Question: What are the differences between the Swing and AWT implementation?

- **AWT :**
+ components are heavyweight, meaning they depend on the native system's GUI.
+ limited to the native system's look and feel.
+ components are rigid and lack advanced features.
+ uses the older, more basic event delegation model.
+ difficult to customize as it relies on native peer components.

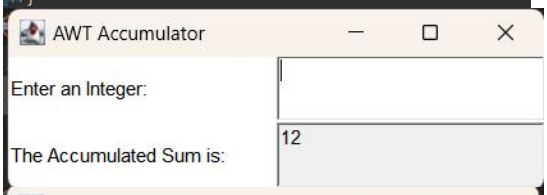
- **Swing:**
+ components are lightweight, meaning they are rendered using Java, independent of the OS.
+ can adopt pluggable look and feel, including Windows, Metal, Nimbus, etc.
+ provides advanced components like JTable, JTree, and JTabbedPane.
+ follows a more advanced event-handling model.
+ highly customizable due to lightweight rendering.

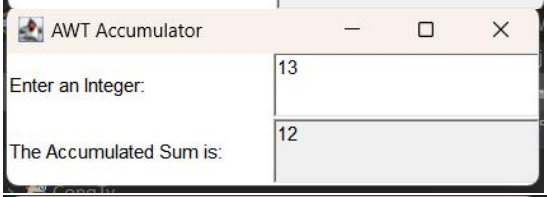
Question: The Aims class must be updated to handle any exceptions generated when the play() methods are called. What happens when you don't update for them to catch?

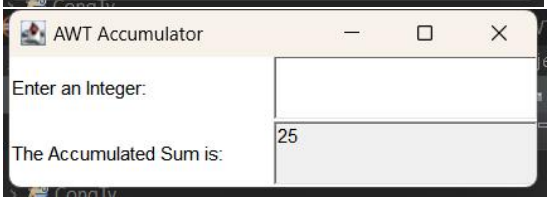
An error will be occurred: "The method play() from the type Playable refers to the missing type PlayerException".

1. Swing components

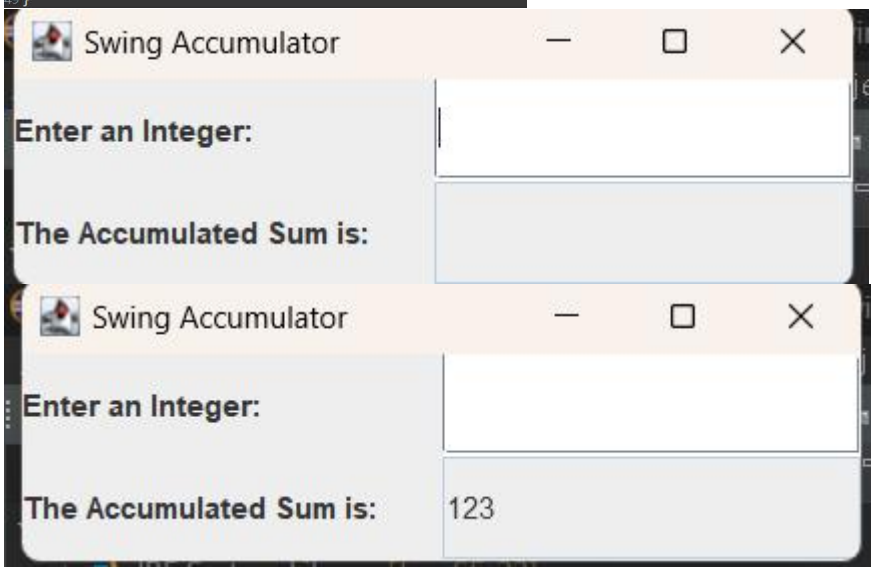
```java
package hust.soict.dsai.swing;

import java.awt.*;
import java.awt.event.*;

public class AWTAccumulator extends Frame{
    private TextField tfInput;
    private TextField tfOutput;
    private int sum= 0;

    public AWTAccumulator() {
        setLayout(new GridLayout(2,2)); //pixels size of the scr

        add(new Label("Enter an Integer: "));

        tfInput= new TextField(10);
        add(tfInput);
        tfInput.addActionListener(new TFInputListener());

        add(new Label("The Accumulated Sum is: "));

        tfOutput= new TextField(10);
        tfOutput.setEditable(false);
        add(tfOutput);

        setTitle("AWT Accumulator");
        setSize(350, 120);
        setVisible(true);
    }

    public static void main(String[] args) {
        new AWTAccumulator();
    }

    private class TFInputListener implements ActionListener {
        @Override
        public void actionPerformed(ActionEvent evt) {
            int numberIn= Integer.parseInt(tfInput.getText());
            sum+= numberIn;
            tfInput.setText("");
            tfOutput.setText(sum+ "");
        }
    }
}
```



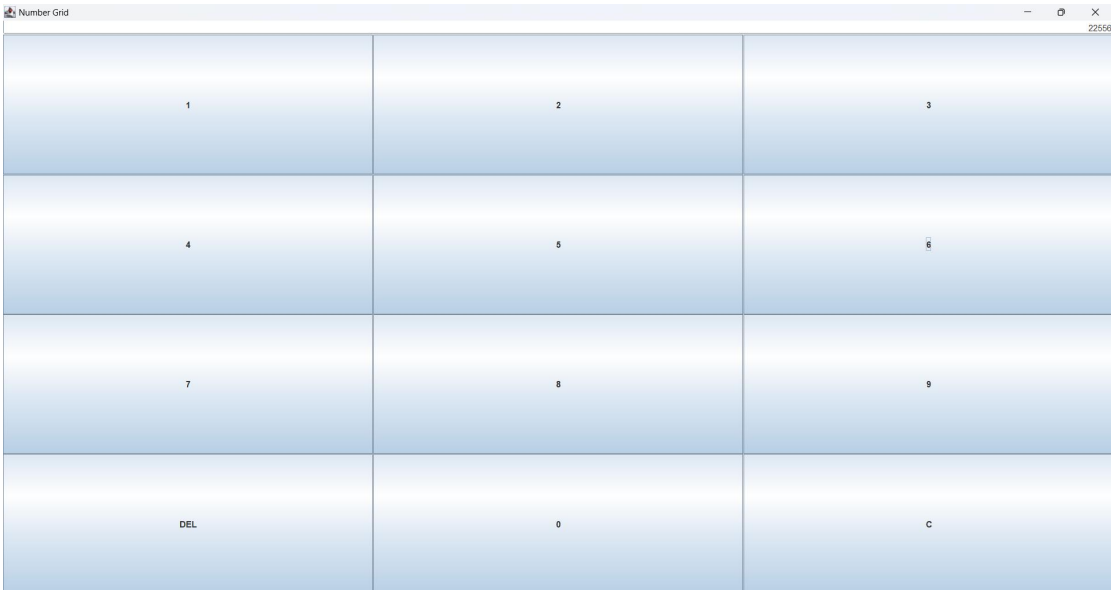




```java
public class SwingAccumulator extends JFrame{
    private JTextField tfInput;
    private JTextField tfOutput;
    private int sum= 0;


    public SwingAccumulator() {
        Container cp= getContentPane();
        cp.setLayout(new GridLayout(2, 2));

        cp.add(new JLabel("Enter an Integer: "));

        tfInput= new JTextField(10);
        cp.add(tfInput);
        tfInput.addActionListener(new TFInputListener());

        cp.add(new JLabel("The Accumulated Sum is: "));

        tfOutput= new JTextField(10);
        tfOutput.setEditable(false);
        cp.add(tfOutput);

        setTitle("Swing Accumulator");
        setSize(350, 120);
        setVisible(true);
    }

    public static void main(String[] args) {
        new SwingAccumulator();
    }

    private class TFInputListener implements ActionListener {
        @Override
        public void actionPerformed(ActionEvent evt) {
            int numberIn= Integer.parseInt(tfInput.getText());
            sum+= numberIn;
            tfInput.setText("");
            tfOutput.setText(sum+ "");
        }
    }
}
```



```java
public class NumberGrid extends JFrame{
    private JButton[] btnNumbers= new JButton[10];
    private JButton btnDelete, btnReset;
    private JTextField tfDisplay;

    public NumberGrid() {

        tfDisplay= new JTextField();
        tfDisplay.setComponentOrientation(
                ComponentOrientation.RIGHT_TO_LEFT);

        JPanel panelButtons= new JPanel( new GridLayout(4, 3));
        addButtons(panelButtons);

        Container cp= getContentPane();
        cp.setLayout(new BorderLayout());
        cp.add(tfDisplay, BorderLayout.NORTH);
        cp.add(panelButtons, BorderLayout.CENTER);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setTitle("Number Grid");
        setSize(200, 200);
        setVisible(true);
    }
    void addButtons(JPanel panelButtons) {
        ButtonListener btnListener= new ButtonListener();
        for(int i=1; i<=9; i++) {
            btnNumbers[i]= new JButton(""+i);
            panelButtons.add(btnNumbers[i]);
            btnNumbers[i].addActionListener(btnListener);
        }

        btnDelete= new JButton("DEL");
        panelButtons.add(btnDelete);
        btnDelete.addActionListener(btnListener);

        btnNumbers[0]= new JButton("0");
        panelButtons.add(btnNumbers[0]);
        btnNumbers[0].addActionListener(btnListener);

        btnReset= new JButton("C");
        panelButtons.add(btnReset);
        btnReset.addActionListener(btnListener);
    }
    public static void main(String[] args) {
        new NumberGrid();
    }

    private class ButtonListener implements ActionListener{
        @Override
        public void actionPerformed(ActionEvent e) {
            String button= e.getActionCommand();
            if(button.charAt(0)>= '0' && button.charAt(0)<= '9') {
                tfDisplay.setText(tfDisplay.getText()+ button);
            }
            else if (button.equals("DEL")){
                //handle "DEL" case;
                if (tfDisplay.getText().length() >= 1) {
                    tfDisplay.setText(tfDisplay.getText().substring(0, tfDisplay.getText().length()-1));
                }
            }
            else {
                //handle "C" case;
                tfDisplay.setText("");
            }
        }
    }
}
```



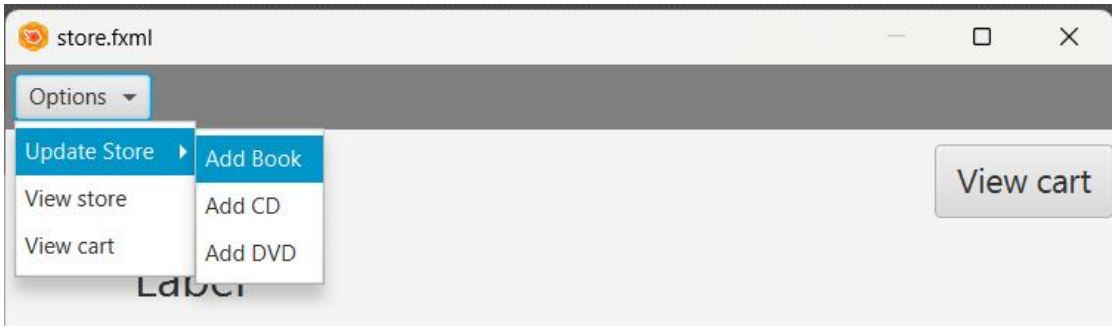
2. AIMS

StoreScreen:

```
public class StoreScreen extends JFrame {
    private Store store;
    private Cart cart;
    private ControllerScreen c;
    JMenuBar createMenuBar() {
        JMenu menu = new JMenu("Option");
        JMenu smUpdateStore = new JMenu("Update Store");
        JMenuItem addBookMenu= new JMenuItem("Add Book");
        addBookMenu.addActionListener(e->{
            c.showAddBookScreen();
        });
        smUpdateStore.add(addBookMenu);
        JMenuItem addCDMenu=new JMenuItem("Add CD");
        addCDMenu.addActionListener(e->{
            c.showAddCDCreen();
        });
        smUpdateStore.add(addCDMenu);
        JMenuItem addDVDMenu= new JMenuItem("Add DVD");
        addDVDMenu.addActionListener(e->{
            c.showAddDVDScreen();
        });
        smUpdateStore.add(addDVDMenu);

        menu.add(smUpdateStore);
        JMenuItem viewStoreMenu= new JMenuItem("View store");
        viewStoreMenu.addActionListener(e->{
            c.showStoreScreen();
        });
        menu.add(viewStoreMenu);

        JMenuItem viewCartMenu= new JMenuItem("View cart");
        viewCartMenu.addActionListener(e->{
            c.showCartScreen();
        });
        menu.add(viewCartMenu);

        JMenuBar menuBar = new JMenuBar();
        menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
        menuBar.add(menu);

        return menuBar;
    }
```

```
    JPanel createHeader() {
        JPanel header = new JPanel();
        header.setLayout(new BoxLayout(header, BoxLayout.X_AXIS));

        JLabel title = new JLabel("AIMS");
        title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 50));
        title.setForeground(Color.CYAN);

        JButton btnCart = new JButton("View cart");

        btnCart.setPreferredSize(new Dimension(100, 50));
        btnCart.setMaximumSize(new Dimension(100, 50));
        btnCart.addActionListener(new ActionListener() {
                @Override
                public void actionPerformed(ActionEvent e) {
                    c.showCartScreen();
                }
        });


        header.add(Box.createRigidArea(new Dimension(10, 10)));
        header.add(title);
        header.add(Box.createHorizontalGlue());
        header.add(btnCart);
        header.add(Box.createRigidArea(new Dimension(10, 10)));

        return header;
    }

    JPanel createNorth() {
        JPanel north = new JPanel();
        north.setLayout(new BoxLayout(north, BoxLayout.Y_AXIS));
        north.add(createMenuBar());
        north.add(createHeader());
        return north;
    }

    JPanel createCenter() {

        JPanel center = new JPanel();
        center.setLayout(new GridLayout(3, 3, 2, 2));

        ArrayList<Media> mediaInStore = store.getItemsInStore();
        for (int i=0; i< mediaInStore.size(); i++) {
            MediaStore cell = new MediaStore(mediaInStore.get(i), cart);
            center.add(cell);
        }

        return center;
    }
```

```
    public StoreScreen(Store store, Cart cart,ControllerScreen c) {
        this.store = store;
        this.cart = cart;
        this.c=c;
        Container cp = getContentPane();
        cp.setLayout(new BorderLayout());

        cp.add(createNorth(), BorderLayout.NORTH);
        cp.add(createCenter(), BorderLayout.CENTER);

//      setVisible(true);
        setTitle("Store");
        setSize(1024, 768);

    }

    public static void main(String[] args) {
    }

}
```



MediaStore:

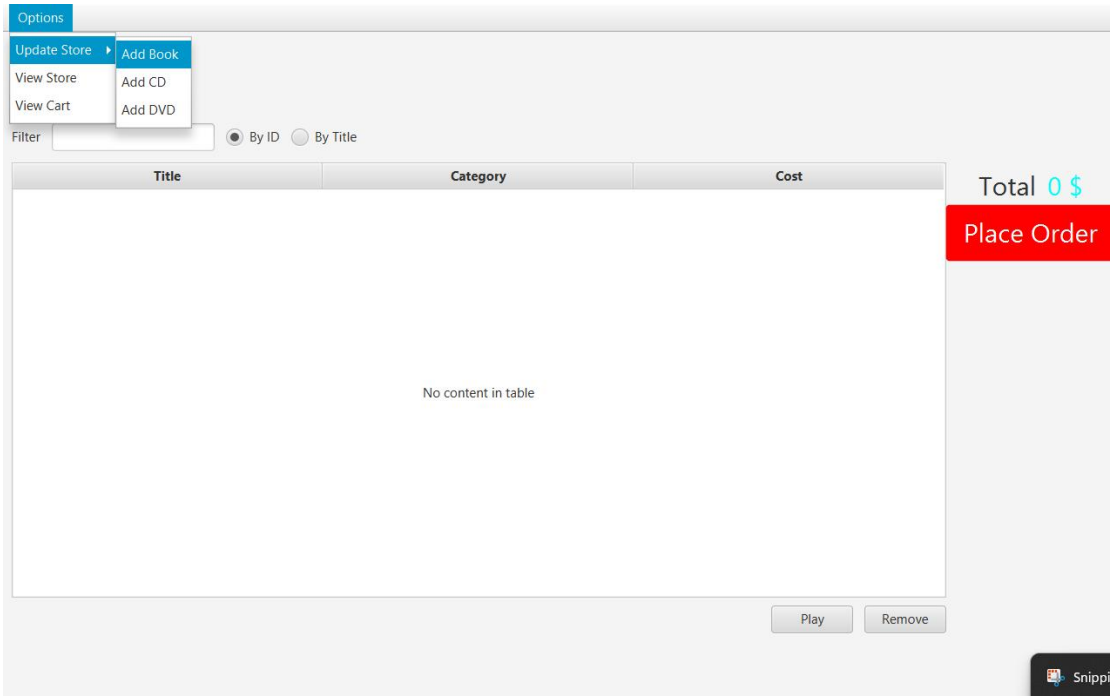
```java
public class MediaStore extends JPanel {

    private Media media;

    public MediaStore(Media media, Cart cart) {
        this.media = media;
        this.setLayout(new BoxLayout(this, BoxLayout.Y_AXIS));

        JLabel title = new JLabel(media.getTitle());
        title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 20));
        title.setAlignmentX(CENTER_ALIGNMENT);

        JLabel cost = new JLabel(""+media.getCost()+"$");
        cost.setAlignmentX(CENTER_ALIGNMENT);

        JPanel container = new JPanel();
        container.setLayout(new FlowLayout(FlowLayout.CENTER));

        JButton addCartBtn = new JButton("Add to Cart");
        addCartBtn.addActionListener(new ActionListener() {

            @Override
            public void actionPerformed(ActionEvent e) {
                try {
                    cart.addMedia(media);
                } catch (LimitExceededException e1) {
                    // TODO Auto-generated catch block
                    e1.printStackTrace();
                }
            }
        });
        container.add(addCartBtn);

     if (media instanceof Playable) {
         JButton playBtn = new JButton("Play");
         playBtn.addActionListener(new ActionListener() {
             @Override
             public void actionPerformed(ActionEvent e) {
                 JDialog playDialog = new JDialog();
                 JPanel mainGui = new JPanel(new BorderLayout());
                 mainGui.setBorder(new EmptyBorder(20, 20, 20, 20));

                 // Display Playing Message
                 mainGui.add(new JLabel("Playing... " + media.getTitle()), BorderLayout.CENTER);
                 System.out.println(media.getTitle());
                 // Close Button
                 JPanel buttonPanel = new JPanel(new FlowLayout());
                 JButton close = new JButton("Stop");
                 close.addActionListener(ev -> {
                     playDialog.setVisible(false);
                     System.out.println("Stopped playing.");
                 });
                 buttonPanel.add(close);
                 mainGui.add(buttonPanel, BorderLayout.SOUTH);

                 playDialog.setContentPane(mainGui);
                 playDialog.setLocationRelativeTo(playBtn);
                 playDialog.pack();

                 // Show Dialog
                 playDialog.setVisible(true);
             }
         });
         container.add(playBtn);
     }

        this.add(Box.createVerticalGlue());
        this.add(title);
        this.add(cost);
        this.add(Box.createVerticalGlue());
        this.add(container);

        this.setBorder(BorderFactory.createLineBorder(Color.BLACK));
```

3. JavaFX API

```java
package hust.soict.dsai.javafx;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

public class Painter extends Application{

    @Override
    public void start(Stage stage) throws Exception {
        Parent root= FXMLLoader.load(getClass().
                getResource("/hust/soict/dsai/javafx/Painter.fxml"));

        Scene scene= new Scene(root);
        stage.setTitle("Painter");
        stage.setScene(scene);
        stage.show();
    }

    public static void main(String[] args) {
        launch(args);
    }
}
```

```java
package hust.soict.dsai.javafx;

import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.RadioButton;
import javafx.scene.control.ToggleGroup;
import javafx.scene.input.MouseEvent;
import javafx.scene.layout.Pane;
import javafx.scene.paint.Color;
import javafx.scene.shape.Circle;


public class PainterController {

    private int color;

    @FXML
    private ToggleGroup identity;

    @FXML
    void chooseOption(ActionEvent event) {
        String button = ((RadioButton)event.getSource()).getText();
        if (button.equals("Pen")) {
            System.out.println("1");
            color = 1;
        }else {
            System.out.println("0");
            color = 0;
        }
    }

    @FXML
    private Pane drawingAreaPane;

    @FXML
    void clearButtonPressed(ActionEvent event) {
        drawingAreaPane.getChildren().clear();
    }

    @FXML
    void drawingAreaMouseDragged(MouseEvent event) {
            Circle newCircle = new Circle(event.getX(), event.getY(), 4.0, Color.WHITE);
            drawingAreaPane.getChildren().add(newCircle);
        }
}
```

4. CartScreen



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

```
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     }
49
50     public static void main(String[] args) {
51     }
```

6+7+8+ 9+ 10+ 11.

```java
public class CartScreenController {

    private Cart cart;
    private ControllerScreen controllerScreen;

    @FXML
    private TableView<Media> tblMedia;

    @FXML
    private TableColumn<Media, String> colMediaCategory;

    @FXML
    private TableColumn<Media, Float> colMediaCost;

    @FXML
    private TableColumn<Media, String> colMediaTitle;

    @FXML
    private ToggleGroup filterCategory;

    @FXML
    private Label totalCost;

    @FXML
    private Button btnPlay;

    @FXML
    private Button btnRemove;

    @FXML
    private Label playingMedia;

    @FXML
    private Button btnStop;

    @FXML
    private Button btnOrder;

    @FXML
    private RadioButton radioBtnFilterId;

    @FXML
    private TextField tfFilter;

    public CartScreenController(Cart cart , ControllerScreen controllerScreen) {
        super();
        this.cart = cart;
        this.controllerScreen = controllerScreen;
    }


    void updateButtonBar(Media media) {
        btnRemove.setVisible(true);
        if (media instanceof Playable) {
            btnPlay.setVisible(true);
        }else {
            btnPlay.setVisible(false);
        }
    }

    void showFilterMedia(String searchString) {
        if(searchString.isEmpty()) {
            tblMedia.setItems(this.cart.getItemsOrdered());
        } else {
            if(radioBtnFilterId.isSelected()) {
                tblMedia.setItems(new FilteredList<Media>(this.cart.getItemsOrdered(),item-> item.getId()==Integer.parseInt(searchString)));
            }else
            tblMedia.setItems(new FilteredList<Media>(this.cart.getItemsOrdered(),item-> item.getTitle().contains(searchString)));
        }
```

```java
    @FXML
    private void initialize() {
        colMediaTitle.setCellValueFactory(
                new PropertyValueFactory<Media, String>("title"));

        colMediaCategory.setCellValueFactory(
                new PropertyValueFactory<Media, String>("category"));
        colMediaCost.setCellValueFactory(
                new PropertyValueFactory<Media, Float>("cost"));

        tblMedia.setItems(this.cart.getItemsOrdered());
        totalCost.setText(cart.totalCost()+"$");
        btnPlay.setVisible(false);
        btnRemove.setVisible(false);
        playingMedia.setVisible(false);
        btnStop.setVisible(false);
        tfFilter.textProperty().addListener(new ChangeListener<String>() {

            @Override
            public void changed(ObservableValue<? extends String> observable, String oldValue, String newValue) {

                showFilterMedia(newValue);
            }
        });
        tblMedia.getSelectionModel().selectedItemProperty().addListener(
                new ChangeListener<Media>() {

                    @Override
                    public void changed(ObservableValue<? extends Media> observable, Media oldValue,
                                        Media newValue) {
                        if(newValue != null) {
                            updateButtonBar(newValue);
                        }
                        totalCost.setText(cart.totalCost()+"$");
                    }
                });
    }

    @FXML
    void btnRemovePressed(ActionEvent event) {
        Media media = tblMedia.getSelectionModel().getSelectedItem();
        cart.removeMedia(media);
        totalCost.setText(cart.totalCost()+"$");
    }

    @FXML
    void btnPlayPressed(ActionEvent event) {
        Media media = tblMedia.getSelectionModel().getSelectedItem();
        playingMedia.setText("Playing "+media.getTitle()+"....");
        playingMedia.setVisible(true);
        btnStop.setVisible(true);
    }

    @FXML
    void btnStopPressed(ActionEvent event) {
        playingMedia.setVisible(false);
        btnStop.setVisible(false);
    }

    @FXML
    void btnOrderPressed(ActionEvent event) {
        System.out.println("Order");
        btnOrder.setText("Success!!!");
        btnOrder.setDisable(true);
        cart.getItemsOrdered().removeAll(cart.getItemsOrdered());
        totalCost.setText("0.0$");
        PauseTransition pt = new PauseTransition(Duration.seconds(1));
        pt.setOnFinished(e ->{
            btnOrder.setDisable(false);
```

```java
    @FXML
    void btnRemovePressed(ActionEvent event) {
        Media media = tblMedia.getSelectionModel().getSelectedItem();
        cart.removeMedia(media);
        totalCost.setText(cart.totalCost()+"$");
    }

    @FXML
    void btnPlayPressed(ActionEvent event) {
        Media media = tblMedia.getSelectionModel().getSelectedItem();
        playingMedia.setText("Playing "+media.getTitle()+"....");
        playingMedia.setVisible(true);
        btnStop.setVisible(true);
    }

    @FXML
    void btnStopPressed(ActionEvent event) {
        playingMedia.setVisible(false);
        btnStop.setVisible(false);
    }

    @FXML
    void btnOrderPressed(ActionEvent event) {
        System.out.println("Order");
        btnOrder.setText("Success!!!");
        btnOrder.setDisable(true);
        cart.getItemsOrdered().removeAll(cart.getItemsOrdered());
        totalCost.setText("0.0$");
        PauseTransition pt = new PauseTransition(Duration.seconds(1));
        pt.setOnFinished(e ->{
            btnOrder.setDisable(false);

            playingMedia.setVisible(false);
            btnPlay.setVisible(false);
            btnOrder.setText("Order");
        } );
        pt.playFromStart();
    }

    @FXML
    void changeToStoreScreen(ActionEvent event) {
        this.controllerScreen.showStoreScreen();
    }
    @FXML
    void changeToAddBookScreen(ActionEvent event) {
        this.controllerScreen.showAddBookScreen();
    }

    @FXML
    void changeToAddCDScreen(ActionEvent event) {
        this.controllerScreen.showAddCDCreen();
    }

    @FXML
    void changeToAddDVDScreen(ActionEvent event) {
        this.controllerScreen.showAddDVDScreen();
    }

    @FXML
    void changeToCartScreen(ActionEvent event) {
        this.controllerScreen.showCartScreen();
    }

    @FXML
    void updateFilter(InputMethodEvent event) {
        System.out.println(event.toString());
    }
```

12.

```java
package hust.soict.dsai.aims.exception;

public class PlayerException extends Exception {

    public PlayerException() {
        // TODO Auto-generated constructor stub
    }

    public PlayerException(String message) {
        super(message);
        // TODO Auto-generated constructor stub
    }

    public PlayerException(Throwable cause) {
        super(cause);
        // TODO Auto-generated constructor stub
    }

    public PlayerException(String message, Throwable cause) {
        super(message, cause);
        // TODO Auto-generated constructor stub
    }

    public PlayerException(String message, Throwable cause, boolean enableSuppression, boolean writableStackTrace) {
        super(message, cause, enableSuppression, writableStackTrace);
        // TODO Auto-generated constructor stub
    }

}
```

```java
public void play() throws PlayerException {
    if(this.getLength()> 0) {
        for(Track track: tracks) {
            track.play();
        }
        System.out.printf("CD's length: %d\n", this.getLength());
    }
    else {
        throw new PlayerException("Error: CD length is non-positive");
    }
}
```

```java
public void play() throws PlayerException {
    if(this.getLength()> 0) {
        System.out.println("Playing DVD: " + this.getTitle());
        System.out.println("DVD length: " + this.getLength());
    }
    else {
        throw new PlayerException("Error: DVD length is non-positive");
    }
}
```

```java
public void play() throws PlayerException {
    if(this.getLength()> 0) {
        System.out.println("Playing track: " + this.getTitle());
        System.out.println("Track length: " + this.getLength());
    }
    else {
        throw new PlayerException("ERROR: CD length is non-positive");
    }
}
```