BÁO CÁO THỰC HÀNH

LAB 05: GUI Programming

Mục lục:

1. Swing Components 2
2. Organizing Swing components with Layout Managers 3
3. Create a graphical user interface for AIMS with Swing 6
   3.1 View Store Screen 6
   3.2 Adding more user interaction 10
4. JavaFX API 11
   4.1 Create FXML file 11
   4.2 Create the Controller class 12
   4.3 Create the application 13
   4.4 Practice exercise 13
5. Setting up the View Cart Screen with ScreenBuilder 14
6. Setting up the View Cart Screen with ScreenBuilder 15
7. View the items in cart – JavaFX's data-driven UI 16
9. Deleting a media 18
10. Filter items in cart – FilteredList 18
11. Complete the AIMS GUI application 18
12. Check all the previous source codes to catch/handle/delegate runtime exceptions 20
   13. Create a class which inherits from Exception 13. 1 Create new class named PlayerException 20
   13.2 Raise the PlayerException in the play() method 21
   13.3 Update play() in the Playable interface 21
   13.4 Update play() in CompactDisc 22
14. Update the Aims class 22
15. Modify the equals() method of Media class 24
16. Reading Document 24
17. Update AIMS class diagram 25

1. Swing Components

- AWTAccumulator Class:

```
package hust.soict.dsai.swing;
import java.awt.Frame;
import java.awt.GridLayout;
import java.awt.Label;
import java.awt.TextField;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class AWTAccumulator extends Frame{
    private final TextField tfInput;
    private final TextField tfOutput;
    private int sum = 0;
    public AWTAccumulator() {
        setLayout(new GridLayout(2, 2));
        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 + "");
        }
    }
}
```

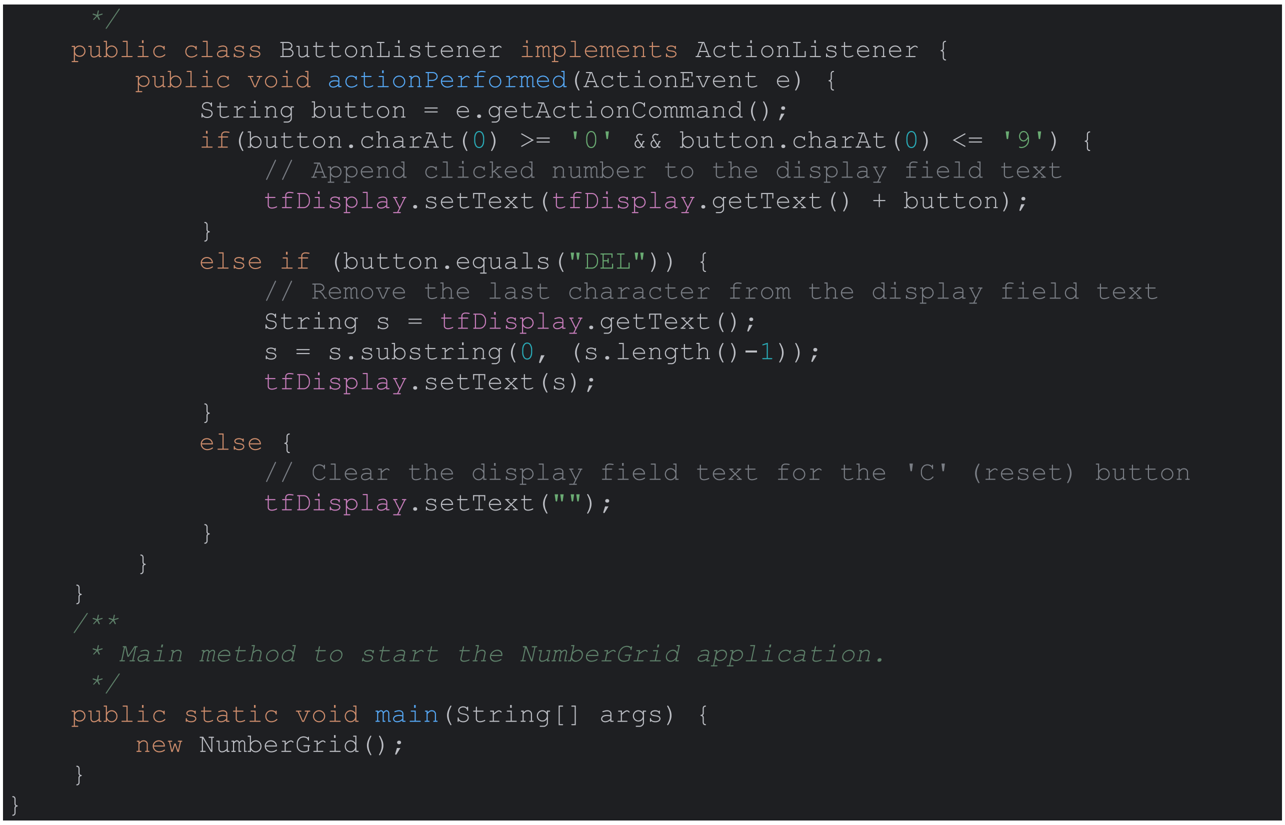
- SwingAccumulator Class:

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

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import static com.sun.glass.ui.Cursor.setVisible;

public class SwingAccumulator extends JFrame {
    private final JTextField tfInput;
    private final JTextField tfOutput;
    private int sum = 0;    //Accumulated sum, init to 0

    //Constructor to setup theGui components and events handlers
    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();
        setSize();
        setVisible(true);
    }

    private void setTitle() {
    }

    private void setSize() {
    }

    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 + "");
        }
    }
}
```

2. Organizing Swing components with Layout Managers

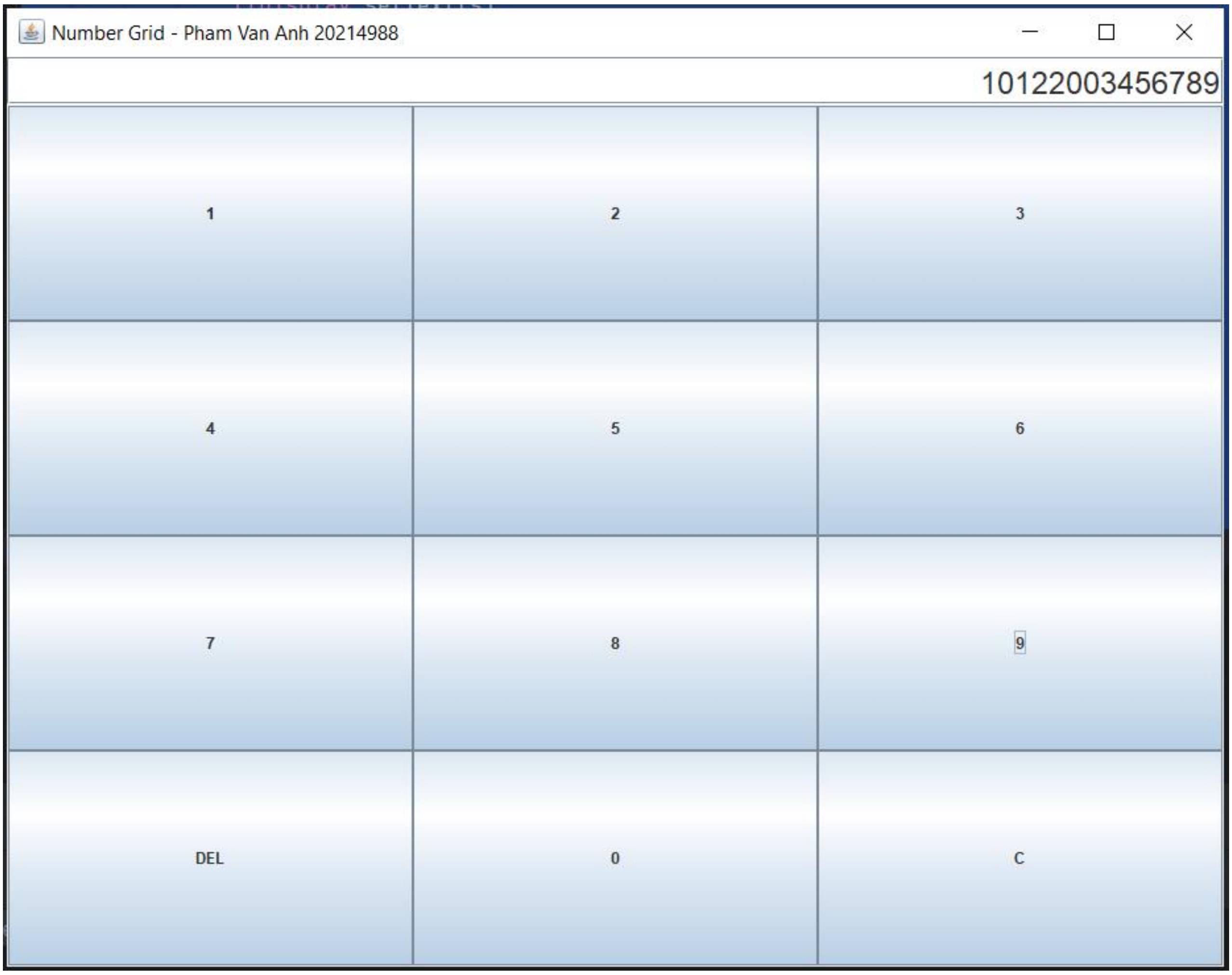
- NumberGrid Class:

```
package hust.soict.dsai.swing;
import java.awt.BorderLayout;
import java.awt.ComponentOrientation;
import java.awt.Container;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JTextField;
/**
 * A simple Swing application that creates a number grid with a display field.
 */
public class NumberGrid extends JFrame {
    private JButton[] btnNumbers = new JButton[10];
    private JButton btnDelete, btnReset;
    private JTextField tfDisplay;
    /**
     * Constructs the NumberGrid frame with a display field and number buttons.
     */
    public NumberGrid() {
        // Create and configure the display field
        tfDisplay = new JTextField();
        tfDisplay.setComponentOrientation(ComponentOrientation.RIGHT_TO_LEFT);
        // Create the panel for number buttons arranged in a 4x3 grid
        JPanel panelButtons = new JPanel(new GridLayout(4, 3));
        addButtons(panelButtons);
        // Get the content pane of the frame and set its layout to BorderLayout
        Container cp = getContentPane();
        cp.setLayout(new BorderLayout());
        // Add the display field to the top (North) of the content pane
        cp.add(tfDisplay, BorderLayout.NORTH);
        // Add the panel with number buttons to the center of the content pane
        cp.add(panelButtons, BorderLayout.CENTER);
        // Set the font size of the display field and configure the frame
        tfDisplay.setFont(tfDisplay.getFont().deriveFont(24.0f));
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setTitle("Number Grid - Pham Van Anh 20214988");
        setSize(500, 400);
        setVisible(true);
    }
    /**
     * Adds number buttons, delete button, and reset button to the specified
panel.
     *
     * @param panelButtons The panel to which buttons will be added.
     */
    void addButtons(JPanel panelButtons) {
        ButtonListener btnListener = new ButtonListener();
        // Add number buttons (1 to 9) to the panel
        for (int i = 1; i <= 9; i++) {
            btnNumbers[i] = new JButton("" + i);
            panelButtons.add(btnNumbers[i]);
            btnNumbers[i].addActionListener(btnListener);
        }
        // Add delete button to the panel
        btnDelete = new JButton("DEL");
        panelButtons.add(btnDelete);
        btnDelete.addActionListener(btnListener);
        // Add '0' button to the panel
        btnNumbers[0] = new JButton("0");
        panelButtons.add(btnNumbers[0]);
        btnNumbers[0].addActionListener(btnListener);
        // Add reset button to the panel
        btnReset = new JButton("C");
        panelButtons.add(btnReset);
        btnReset.addActionListener(btnListener);
    }
    /**
     * ActionListener implementation for handling button clicks.
```



```
     */
    public class ButtonListener implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            String button = e.getActionCommand();
            if(button.charAt(0) >= '0' && button.charAt(0) <= '9') {
                // Append clicked number to the display field text
                tfDisplay.setText(tfDisplay.getText() + button);
            }
            else if (button.equals("DEL")) {
                // Remove the last character from the display field text
                String s = tfDisplay.getText();
                s = s.substring(0, (s.length()-1));
                tfDisplay.setText(s);
            }
            else {
                // Clear the display field text for the 'C' (reset) button
                tfDisplay.setText("");
            }
        }
    }
    /**
     * Main method to start the NumberGrid application.
     */
    public static void main(String[] args) {
        new NumberGrid();
    }
}
```

- Kết quả:



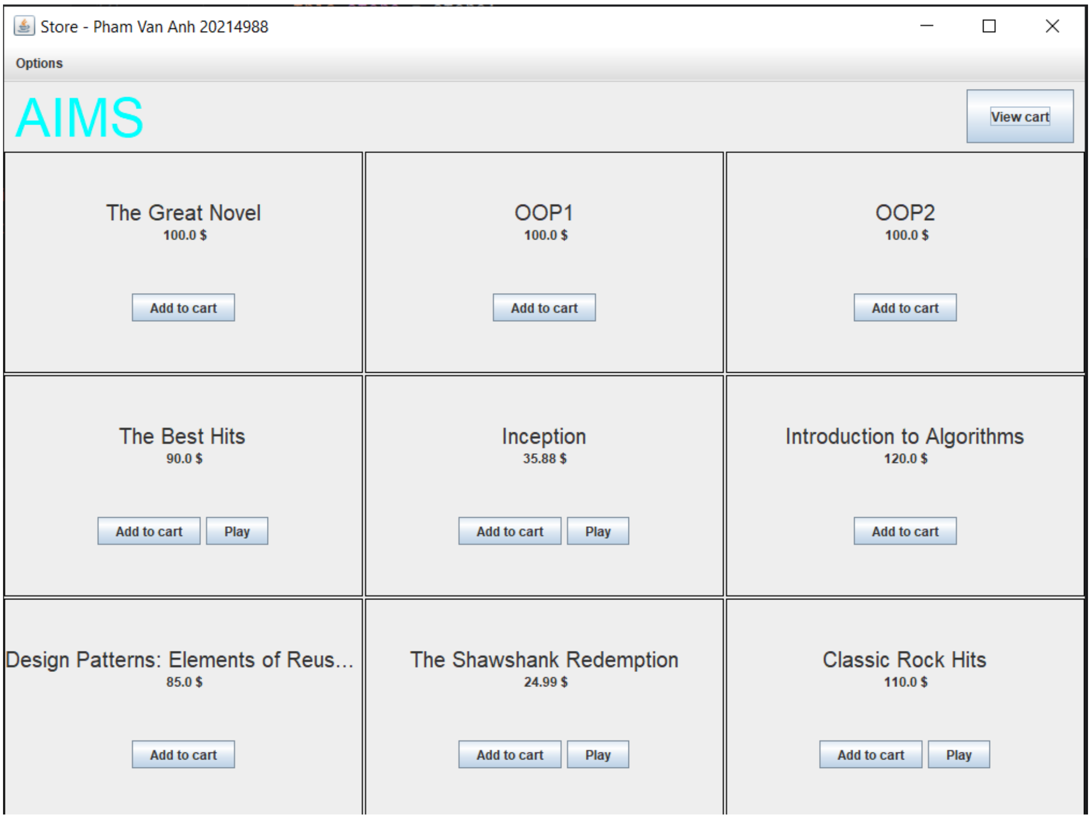
3. Create a graphical user interface for AIMS with Swing

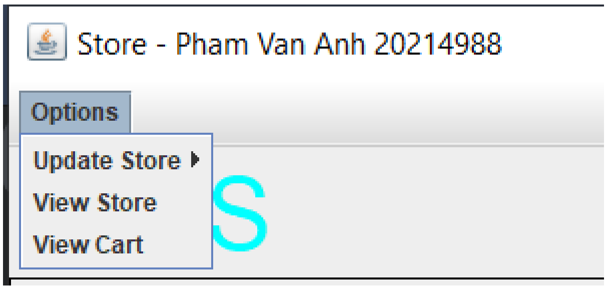
3.1 View Store Screen

- StoreScreen Class:

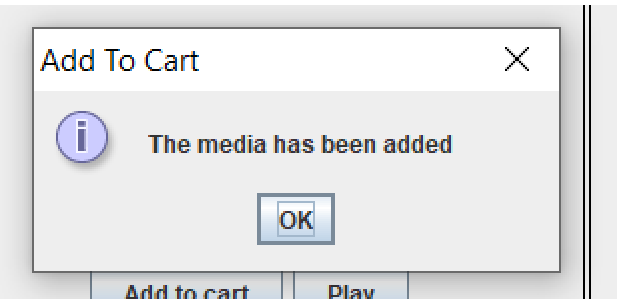
```
package hust.soict.dsai.aims.screen;

import hust.soict.dsai.aims.media.*;
import hust.soict.dsai.aims.store.Store;
import javafx.collections.ObservableList;
import javafx.scene.media.Media;

import javax.naming.LimitExceededException;
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;
import java.util.Arrays;

public class StoreScreen extends JFrame{
    private Store store;
    private JButton btnAddToCart, btnPlay;
    JPanel createNorth() {
        JPanel north = new JPanel();
        north.setLayout(new BoxLayout(north, BoxLayout.Y_AXIS));
        north.add(createMenuBar());
        north.add(createHeader());
        return north;
    }

    JMenuBar createMenuBar() {
        JMenu menu = new JMenu("Options");

        JMenu smUpdateStore = new JMenu("Update Store");
        smUpdateStore.add(new JMenuItem("Add Book"));
        smUpdateStore.add(new JMenuItem("Add CD"));
        smUpdateStore.add(new JMenuItem("Add DVD"));

        menu.add(smUpdateStore);
        menu.add(new JMenuItem("View Store"));
        menu.add(new JMenuItem("View Cart"));

        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 cart = new JButton("View cart");
        cart.setPreferredSize(new Dimension(100, 50));
        cart.setMaximumSize(new Dimension(100, 50));

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

        return header;
    }

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

        ObservableList<Media> mediaInStore = store.getItemsInStore();
        for(Media m: mediaInStore) {
            MediaStore cell = new MediaStore(m);
            center.add(cell);
        }

        return center;
    }

    public class MediaStore extends JPanel {
        private Media media;
        ButtonListener btnListener = new ButtonListener();

        public MediaStore(Media media) {
            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));

            btnAddToCart = new JButton("Add to cart");
            container.add(btnAddToCart);
            btnAddToCart.addActionListener(btnListener);
            if(media instanceof Playable) {
                btnPlay = new JButton("Play");
                container.add(btnPlay);
                btnPlay.addActionListener(btnListener);
            }

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

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

    public class ButtonListener implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            String button = e.getActionCommand();
            if(button.charAt(0) == 'A') {
                JFrame frame = new JFrame("JOptionPane ");
                JOptionPane.showMessageDialog(frame,
                        "The media has been added",
                        "Add To Cart",
                        JOptionPane.INFORMATION_MESSAGE);
            }
            else if (button.charAt(0) == 'P') {
                JFrame frame = new JFrame("JOptionPane ");
                JOptionPane.showMessageDialog(frame,
                        "Play The Media",
                        "Play",
                        JOptionPane.INFORMATION_MESSAGE);
            }
        }
    }

    public StoreScreen(Store store) {
        this.store = store;
        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) throws LimitExceededException {
        Store store = new Store();
        // Book
        Book initBook = new Book("The Great Novel", "Book", 100.00f,
                Arrays.asList("John Doe", "Jane Smith"));

        // Book
        Book initBook2 = new Book("OOP1", "Book", 100.00f,
                Arrays.asList("xyz", "abc"));

        // Book
        Book initBook3 = new Book("OOP2", "Book", 100.00f,
                Arrays.asList("ghf", "sak"));

        // DVD
        DigitalVideoDisc initDVD = new DigitalVideoDisc("Inception", "DVD",
                "Christopher Nolan", 150, 35.88f);

        // CD
        ArrayList<Track> tracks = new ArrayList<>();
        tracks.add(new Track("Song 1", 3));
        tracks.add(new Track("Song 2", 4));
        CompactDisc initCD = new CompactDisc("The Best Hits", "CD", 90.00f,
                "Various Artists", tracks);

        store.addMedia(initBook);
        store.addMedia(initBook2);
        store.addMedia(initBook3);
        store.addMedia(initCD);
        store.addMedia(initDVD);

        // Additional Books
        Book additionalBook1 = new Book("Introduction to Algorithms", "Book",
120.00f,
                Arrays.asList("Thomas H. Cormen", "Charles E. Leiserson",
"Ronald L. Rivest", "Clifford Stein"));

        Book additionalBook2 = new Book("Design Patterns: Elements of Reusable
Object-Oriented Software", "Book", 85.00f,
                Arrays.asList("Erich Gamma", "Richard Helm", "Ralph Johnson",
"John Vlissides"));

        // Additional DVD
        DigitalVideoDisc additionalDVD = new DigitalVideoDisc("The Shawshank
Redemption", "DVD",
                "Frank Darabont", 142, 24.99f);

        // Additional CD
        ArrayList<Track> additionalTracks = new ArrayList<>();
        additionalTracks.add(new Track("Bohemian Rhapsody", 6));
        additionalTracks.add(new Track("Hotel California", 5));
        CompactDisc additionalCD = new CompactDisc("Classic Rock Hits", "CD",
110.00f,
                "Various Artists", additionalTracks);

        store.addMedia(additionalBook1);
        store.addMedia(additionalBook2);
        store.addMedia(additionalDVD);
        store.addMedia(additionalCD);

        new StoreScreen(store);
    }
}
```

- Kết quả:

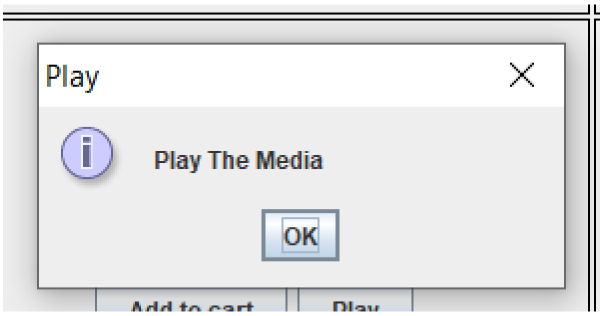




3.2 Adding more user interaction



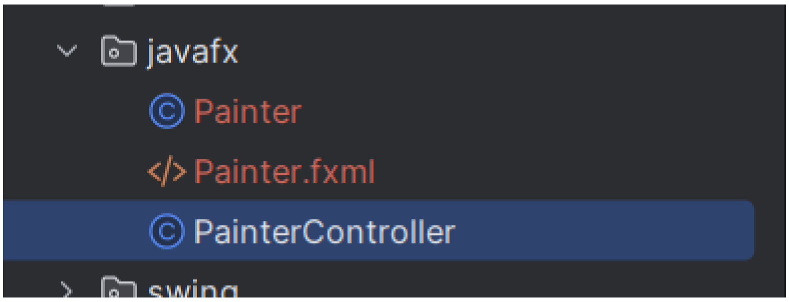
The media has been added



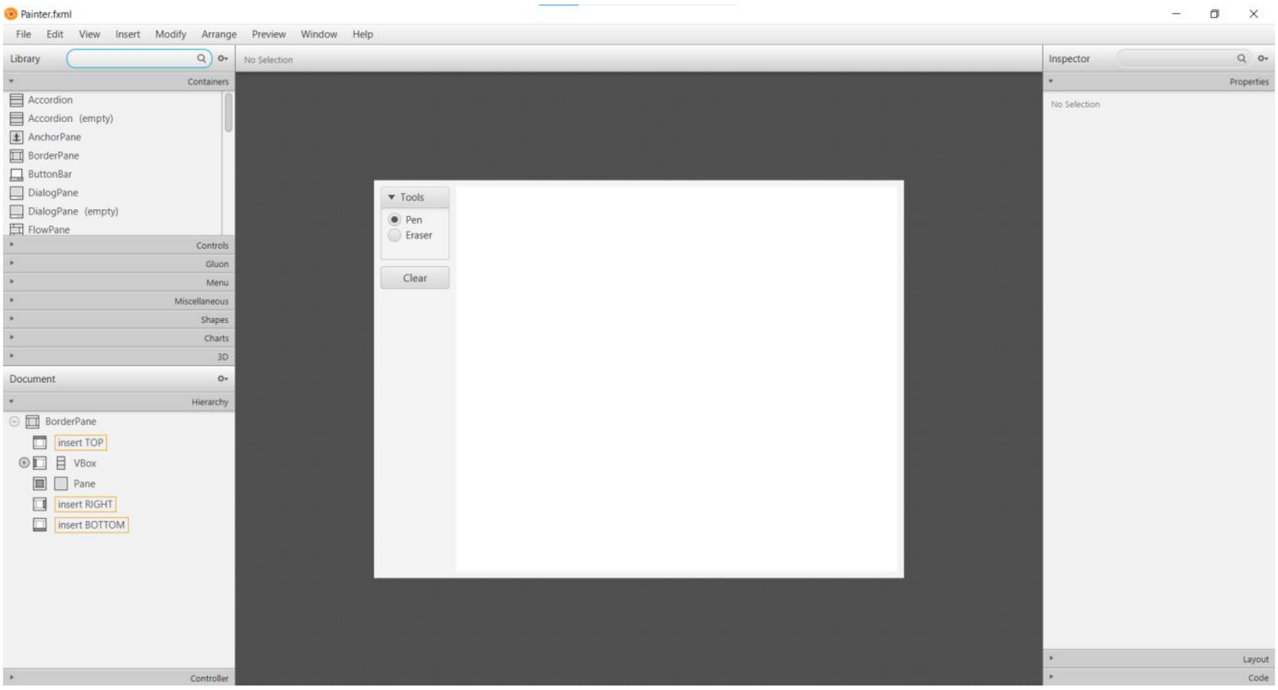
Play

Play The Media

4. JavaFX API



4.1 Create FXML file



4.2 Create the Controller class

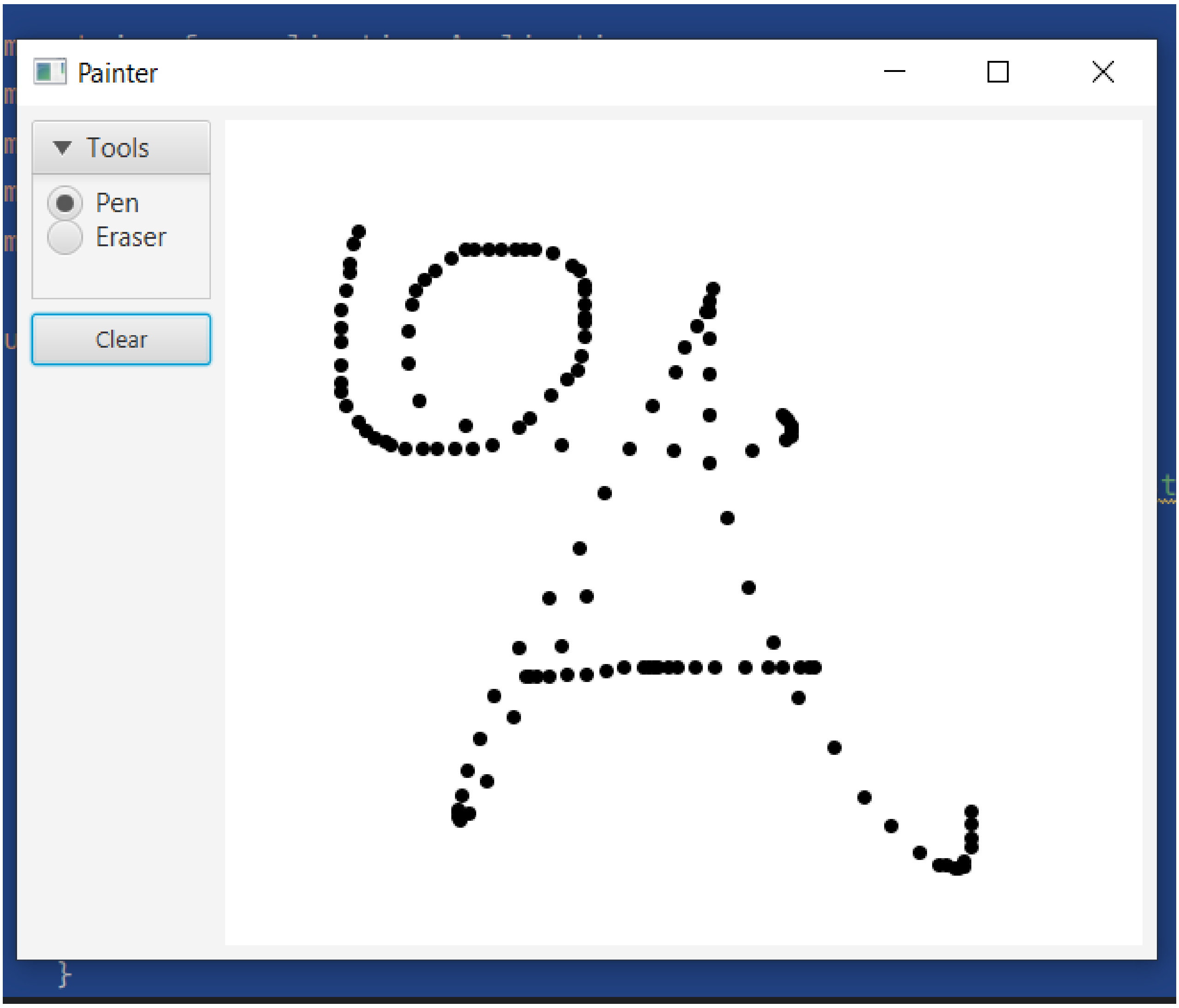
```
package hust.soict.dsai.javafx;

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 {
    @FXML
    private RadioButton ButtonPressed1;

    @FXML
    private RadioButton ButtonPressed2;

    @FXML
    private Pane drawingAreaPane;

    @FXML
    void Selected(ActionEvent event) {
        ToggleGroup question= new ToggleGroup();
        ButtonPressed1.setToggleGroup(question);
        ButtonPressed2.setToggleGroup(question);
        if(ButtonPressed1.isSelected()) {
            ButtonPressed2.setSelected(false);
        }
        else {
            ButtonPressed1.setSelected(false);
        }
    }

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

    @FXML
    void drawingAreaMouseDragged(MouseEvent event) {
        if(ButtonPressed1.isSelected()) {
            Circle newCircle = new Circle(event.getX(), event.getY(), 4,
Color.BLACK);
            drawingAreaPane.getChildren().add(newCircle);
        }
        else {
            Circle newCircle = new Circle(event.getX(), event.getY(), 4,
Color.WHITE);
            drawingAreaPane.getChildren().add(newCircle);
        }
    }

}
```

4.3 Create the application

Painter class:

```
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);
    }
}
```

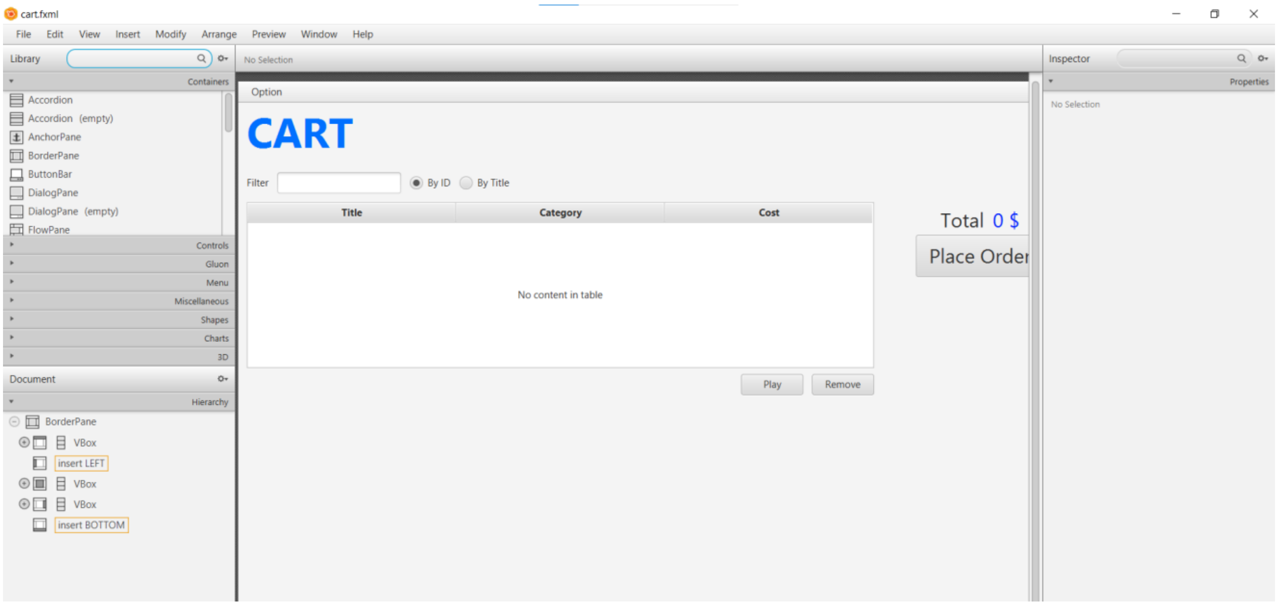
4.4 Practice exercise



5. Setting up the View Cart Screen with

ScreenBuilder

- Cart.fxml:



6. Setting up the View Cart Screen with ScreenBuilder

- CartScreen class:

```
- package hust.soict.dsai.aims.screen;

  import hust.soict.dsai.aims.cart.Cart;
  import javafx.application.Platform;
  import javafx.embed.swing.JFXPanel;
  import javafx.fxml.FXMLLoader;
  import javafx.scene.Parent;
  import javafx.scene.Scene;
  import javax.swing.*;
  import java.awt.*;
  import java.io.IOException;


  public class CartScreen extends JFrame {
      public CartScreen(Cart cart) {
          super();
          JFXPanel fxPanel = new JFXPanel();
          this.add(fxPanel);
          this.setTitle("Cart");
          this.setVisible(true);
          this.setSize(new Dimension(1024, 768));
          this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

          Platform.runLater(new Runnable() {
              @Override
              public void run() {
                  // TODO Auto-generated method stub
```

```
                try {
                    FXMLLoader loader = new
FXMLLoader(getClass().getResource("/hust/soict/dsai/aims/screen/cart.
fxml"));
                    CartScreenController controller = new
CartScreenController(cart);
                    loader.setController(controller);
                    Parent root = loader.load();
                    Scene scene = new Scene(root);
                    fxPanel.setScene(scene);
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

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

- CartScreenController class:

```
public class CartScreenController {
    private Cart cart;
    @FXML
    private TableView<Media> tblMedia;
    @FXML
    private TableColumn<Media, String> colMediaTitle;
    @FXML
    private TableColumn<Media, String> colMediacategory;
    @FXML
    private TableColumn<Media, Float> colMediaCost;
    public CartScreenController (Cart cart) { super();
        this.cart = cart;
    }
    @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());
```

- Cart class:

```
public class Cart {
    public static final int MAX_NUMBERS_ORDERED = 20;
    private ObservableList<Media> itemsOrdered =
FXCollections.observableArrayList();

    public ObservableList<Media> getItemsOrdered() {
        // TODO Auto-generated method stub
        return itemsOrdered;
    }
}
```

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

- CartScreenController Class

Initialize:

```
@FXML
private void initialize() {
    // Show Cart in the Table
    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(Float.toString(cart.totalCost()) + "$");

    // Set Default the button Play and Remove
    btnPlay.setVisible(false);
    btnRemove.setVisible(false);

    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(Float.toString(cart.totalCost()) + "$");
        }
    });
```

- updateBottonBar:

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

9. Deleting a media

- Remove Button:

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

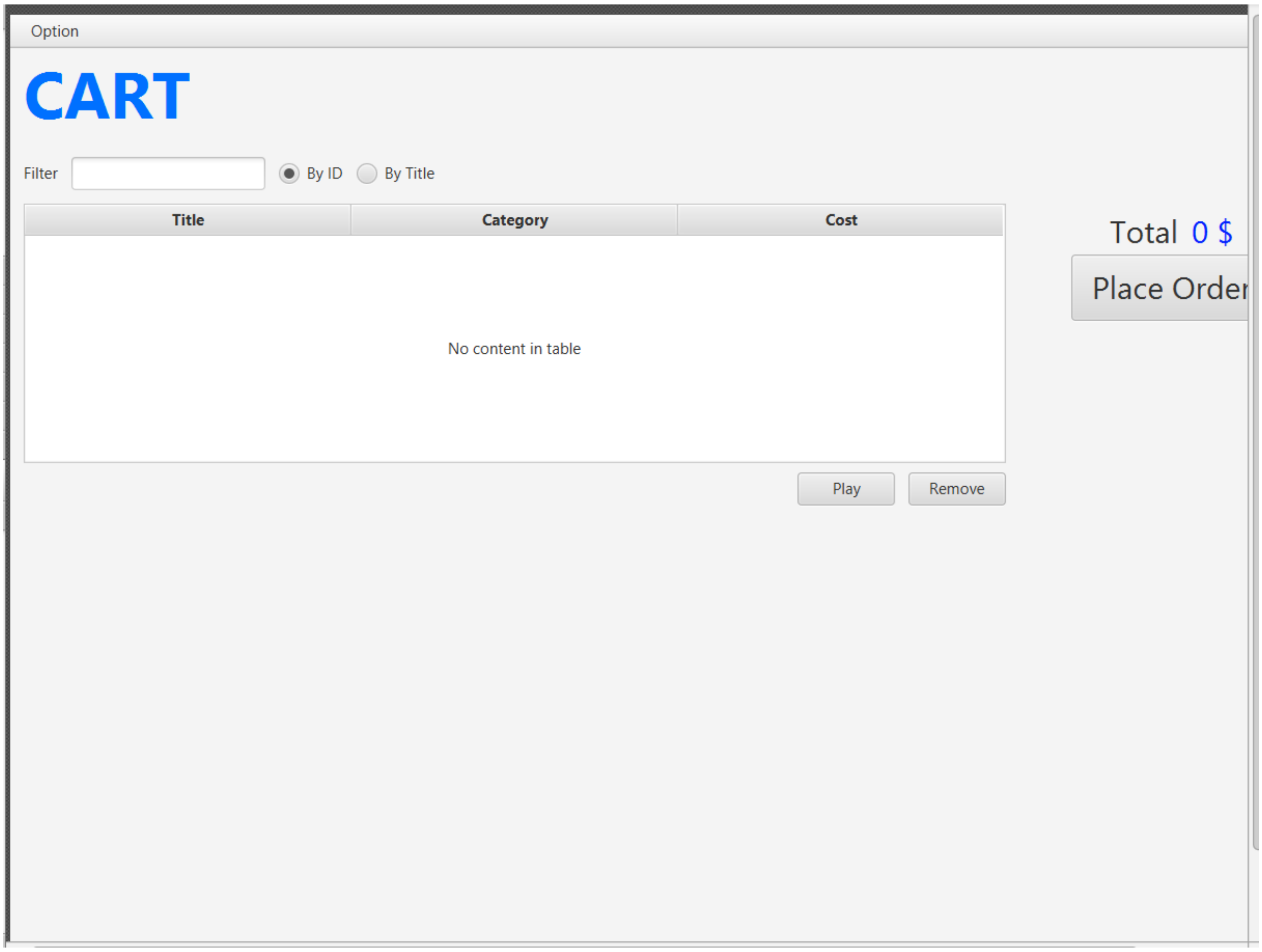
10. Filter items in cart – FilteredList

- Filter list:

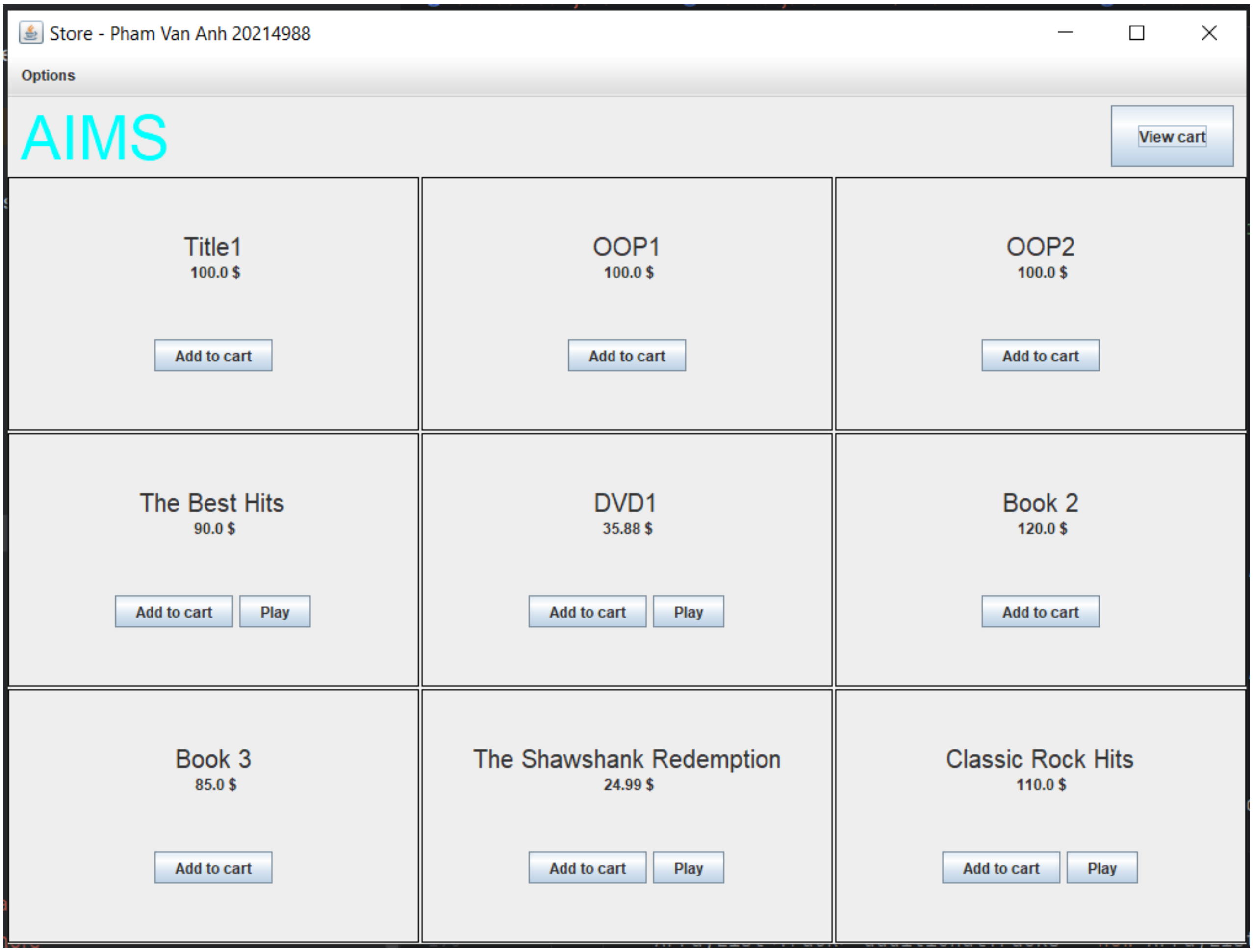
```
// Filter
tfFilter.textProperty().addListener(new ChangeListener<String>() {
    @Override
    public void changed(ObservableValue<? extends String> arg0, String
oldValue, String newValue) {
        showFilteredMedia(newValue);
    }
```

11. Complete the AIMS GUI application

- Cart Screen:



- Store Screen:



Store - Pham Van Anh 20214988

Title1
100.0 $

OOP1
100.0 $

OOP2
100.0 $

The Best Hits
90.0 $

DVD1
35.88 $

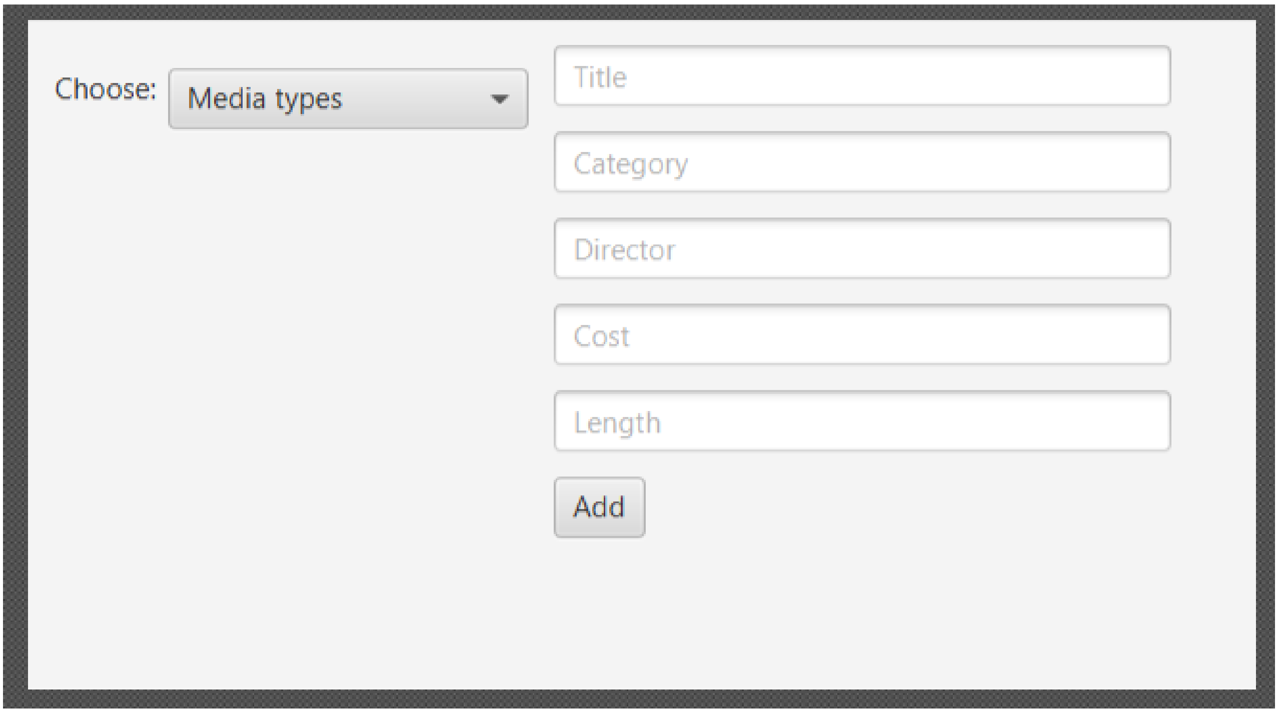
Book 2
120.0 $

Book 3
85.0 $

The Shawshank Redemption
24.99 $

Classic Rock Hits
110.0 $

- Update Store Screen:



Choose: Media types

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

- Cart class:

```
// Add Media to Cart
public void addMedia(Media m) throws LimitExceededException {
    int size = itemsOrdered.size();
    if(size < MAX_NUMBERS_ORDERED) {
        itemsOrdered.add(m);
        System.out.println("The media has been added");
    }
    else {
        throw new LimitExceededException("ERROR: The number of media has
reached its limit");
    }
}
```

13. Create a class which inherits from Exception

13. 1 Create new class named PlayerException

- PlayException class:

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

public class PlayerException extends Exception {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
    }

    public PlayerException() {
        super();
        // 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
    }

    public PlayerException(String message, Throwable cause) {
        super(message, cause);
        // 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
    }
}
```

13.2 Raise the PlayerException in the play() method

- Play trong DigitalVideoDisc Class

```
 @Override
    public void play() throws PlayerException {
        // TODO Auto-generated method stub
        if (this.getLength() > 0) {
            JDialog dialog = new JDialog();
            dialog.setSize(300, 200);

            // create Label
            JLabel text = new JLabel("DVD - Title : " + this.getTitle() + " 
Length : " + this.getLength());
            dialog.add(text);
            dialog.setTitle("Play DVD");
            dialog.setVisible(true);
        } else
            throw new PlayerException("ERROR : DVD length is non-positive");
    }
}
```

- Play trong Track

```
@Override
public void play() throws PlayerException {
    // TODO Auto-generated method stub
    if (this.getLength() > 0) {
        JDialog dialog = new JDialog();
        dialog.setSize(300, 200);

        // create Label
        JLabel text = new JLabel("Track - Title : " + this.getTitle() + " 
Length : " + this.getLength());
        dialog.add(text);
        dialog.setTitle("Play Track");
        dialog.setVisible(true);
    } else
        throw new PlayerException("ERROR : Track length is non-positive");
}
```

13.3 Update play() in the Playable interface

- Play trong Playable

```
package hust.soict.dsai.aims.media;

import hust.soict.dsai.aims.exception.PlayerException;

public interface Playable {
    public void play() throws PlayerException;
}
```

13.4 Update play() in CompactDisc

- Play trong CompactDisc

```
@Override
public void play() throws PlayerException {
    int size = tracks.size();
    JPanel layout = new JPanel(new GridLayout(size, 1));
    if (this.getLength() < 0) {
        throw new PlayerException("ERROR : CD length is non-positive");
    }
    for (Track track : tracks) {
        if (track.getLength() > 0) {
            // create Label
            JLabel text = new JLabel("CD - Title : " + track.getTitle() + "
Length : " + track.getLength());

            layout.add(text);
        } else
            throw new PlayerException("ERROR : Disc length is non-positive");
    }
    JDialog dialog = new JDialog();
    dialog.setSize(300, 200);
    dialog.add(layout);
    dialog.setTitle("Play CD");
    dialog.setVisible(true);
}
```

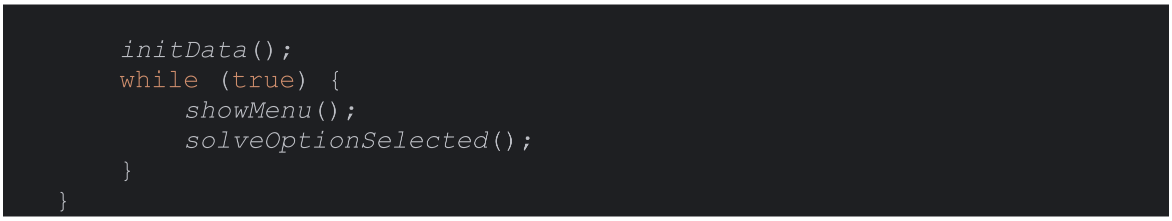
14. Update the Aims class

- Play media:

```
public static void playMedia(Media m) throws PlayerException {
    try {
        if (m instanceof DigitalVideoDisc) {
            System.out.println("\n%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%");
            ((DigitalVideoDisc) m).play();
            System.out.println("%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%\n");
        } else if (m instanceof CompactDisc) {
            System.out.println("\n%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%");
            ((CompactDisc) m).play();
            System.out.println("%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%\n");
        } else {

System.out.println("======================================================");
            System.out.println("===>>>This type of media does not have play
mode<<<===");

System.out.println("======================================================");
        }

    } catch (PlayerException e) {
        handleException(e);
    }
}
```

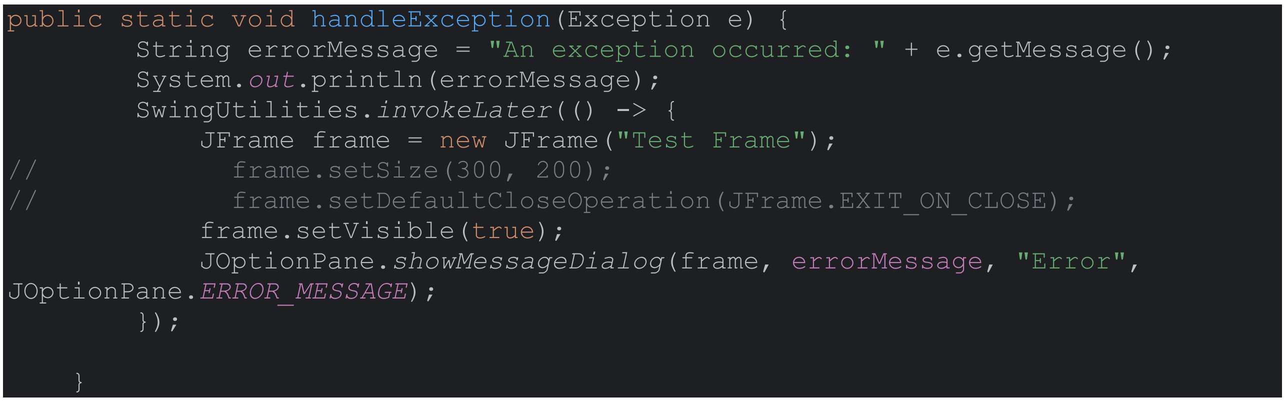
- Main:

```
- public static void main(String[] args) throws PlayerException,
  LimitExceededException {
```



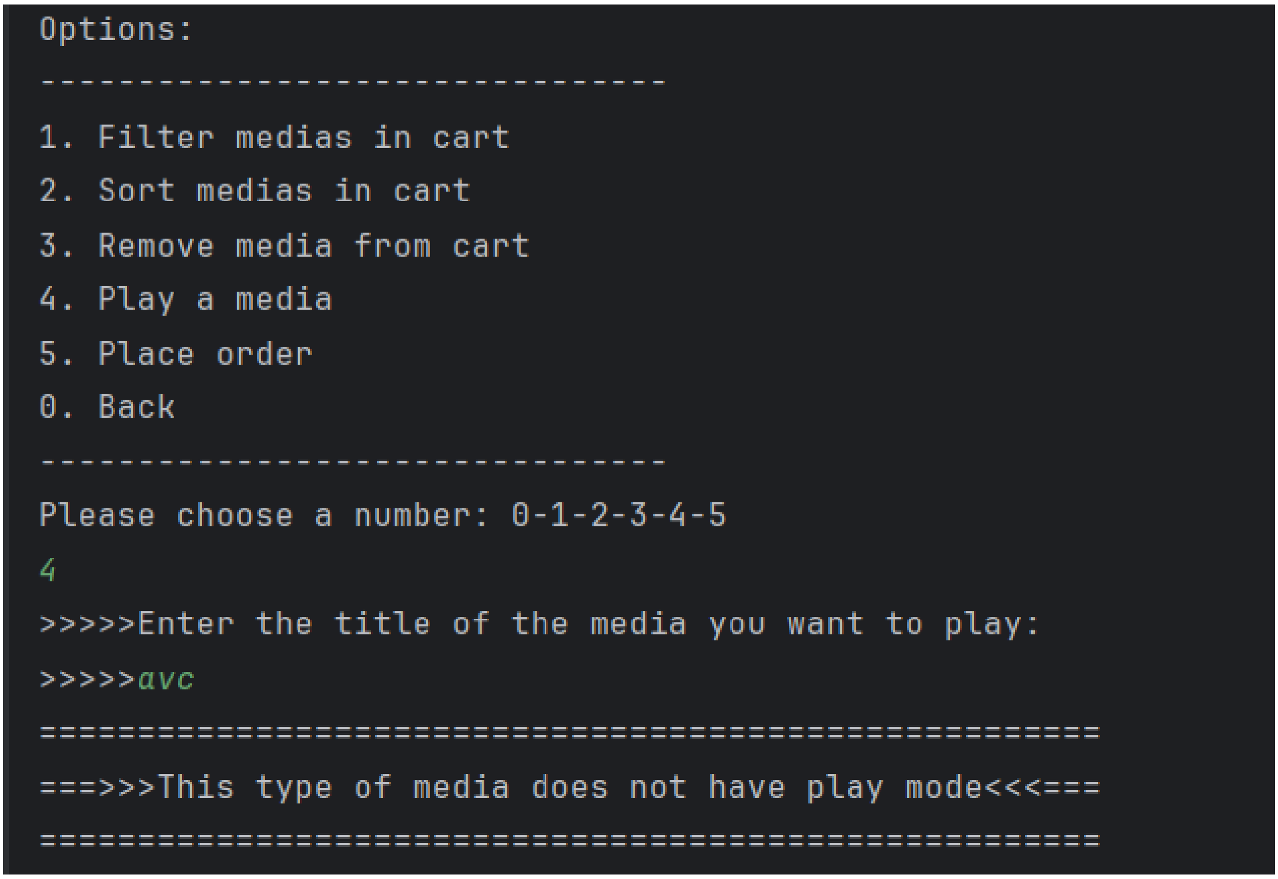
```
        initData();
        while (true) {
            showMenu();
            solveOptionSelected();
        }
    }
```

- Exception:



```
public static void handleException(Exception e) {
        String errorMessage = "An exception occurred: " + e.getMessage();
        System.out.println(errorMessage);
        SwingUtilities.invokeLater(() -> {
            JFrame frame = new JFrame("Test Frame");
//              frame.setSize(300, 200);
//              frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            frame.setVisible(true);
            JOptionPane.showMessageDialog(frame, errorMessage, "Error",
JOptionPane.ERROR_MESSAGE);
        });

    }
```

- Kết quả:



```
Options:
--------------------------------
1. Filter medias in cart
2. Sort medias in cart
3. Remove media from cart
4. Play a media
5. Place order
0. Back
--------------------------------
Please choose a number: 0-1-2-3-4-5
4
>>>>>Enter the title of the media you want to play:
>>>>>avc
======================================================
===>>>This type of media does not have play mode<<<===
======================================================
```

15.Modify the equals() method of Media class

- Equals trong Media class:

```java
@Override
public boolean equals(Object obj) {
    // Check if the object is compared to itself
    if (this == obj) {
        return true;
    }

    // Check if the object is null
    if (obj == null) {
        return false;
    }

    // Check if the object is an instance of Media class
    if (!(obj instanceof Media)) {
        return false;
    }

    // Cast the object to Media type
    Media otherMedia = (Media) obj;

    // Check if the titles are equal
    if (this.getTitle() == null && otherMedia.getTitle() == null) {
        return true;
    } else if (this.getTitle() == null || otherMedia.getTitle() == null) {
        return false;
    } else {
        return this.getTitle().equals(otherMedia.getTitle());
    }
}
```

16.Reading Document

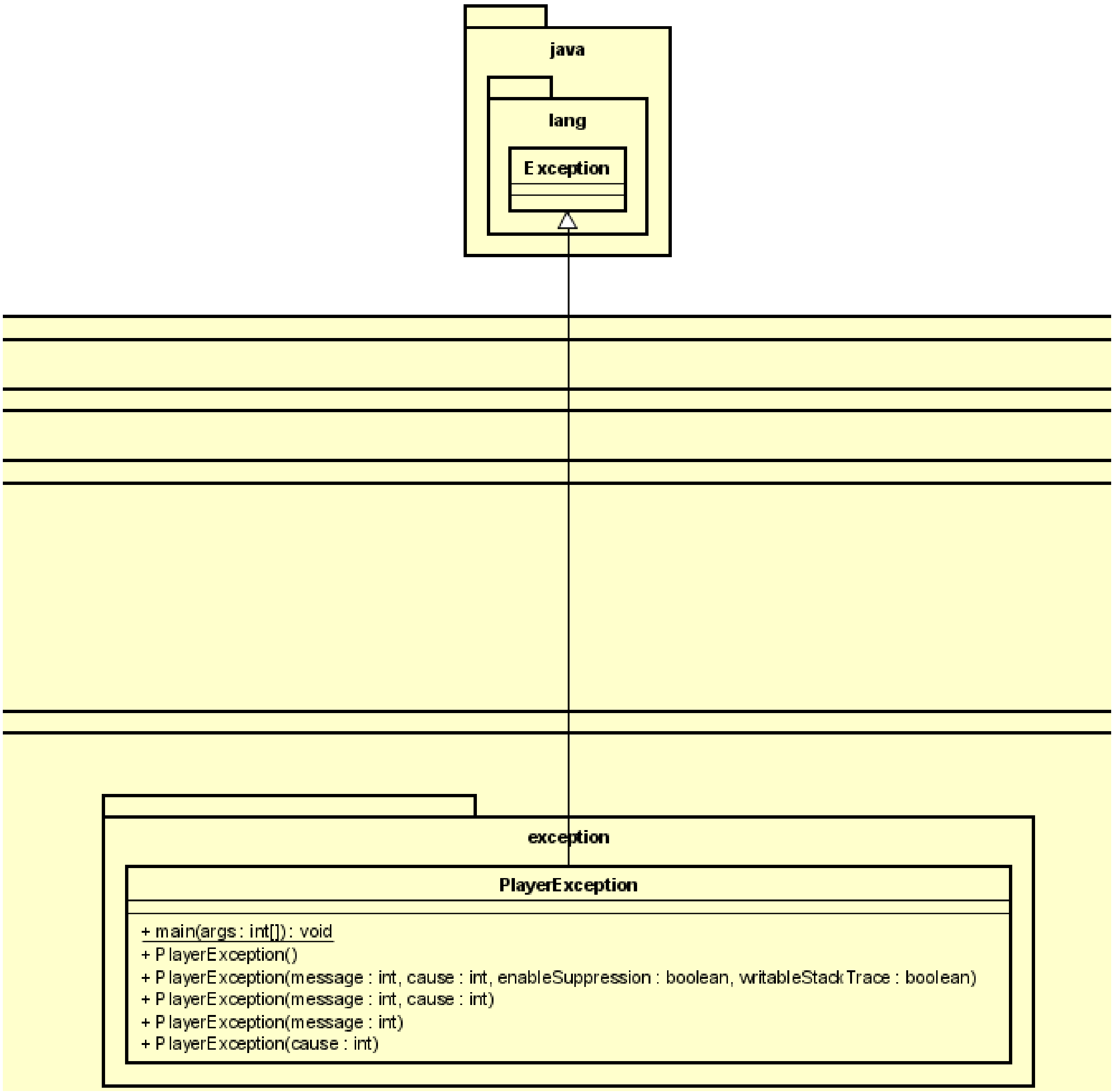
- Exception-handling basics:

https://developer.ibm.com/tutorials/j-perry-exceptions/

- Basic guidelines: Although the examples are in C++, the ideas are important.

https://docs.microsoft.com/en-us/cpp/cpp/errors-and-exception-handling-modern-cpp?view=vs-2019#basic-guidelines

17.Update AIMS class diagram


java
lang
Exception
exception
PlayerException
+ main(args : int[]) : void
+ PlayerException()
+ PlayerException(message : int, cause : int, enableSuppression : boolean, writableStackTrace : boolean)
+ PlayerException(message : int, cause : int)
+ PlayerException(message : int)
+ PlayerException(cause : int)