

Lab 05 – Lê Minh Nghĩa - 20225894

1. Swing Components

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

import ...

public class AWTAccumulator extends Frame {
    private TextField tfInput;    5 usages
    private TextField tfOutput;  4 usages
    private int sum = 0;    2 usages

    public AWTAccumulator() {    1 usage
        setLayout(new GridLayout( rows: 2, cols: 2));

        add(new Label( text: "Enter an Integer: "));
        tfInput = new TextField( columns: 10);
        add(tfInput);
        tfInput.addActionListener(new TfInputListener());

        add(new Label( text: "The Accumulated Sum is: "));
        tfOutput = new TextField( columns: 10);
        tfOutput.setEditable(false);
        add(tfOutput);

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

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

    private class TfInputListener implements ActionListener {    1 usage
        @Override
        public void actionPerformed(ActionEvent e) {
            sum += Integer.parseInt(tfInput.getText());
            tfInput.setText("");
            tfOutput.setText(String.valueOf(sum));
        }
    }
}
```

```

package hust.soict.dsai.swing;

> import ...

public class SwingAccumulator extends JFrame {
    private JTextField tfInput; 5 usages
    private JTextField tfOutput; 4 usages
    private int sum = 0; 2 usages

    public SwingAccumulator() { 1 usage
        Container cp = getContentPane();
        cp.setLayout(new GridLayout( rows: 2, cols: 2));

        cp.add(new JLabel( text: "Enter an Integer: "));
        tfInput = new JTextField( columns: 10);
        cp.add(tfInput);
        tfInput.addActionListener(new TfInputListener());

        cp.add(new JLabel( text: "The Accumulated Sum is: "));
        tfOutput = new JTextField( columns: 10);
        tfOutput.setEditable(false);
        cp.add(tfOutput);

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

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

    private class TfInputListener implements ActionListener { 1 usage
        @Override
        public void actionPerformed(ActionEvent e) {
            sum += Integer.parseInt(tfInput.getText());
            tfInput.setText("");
            tfOutput.setText(String.valueOf(sum));
        }
    }
}

```

2. Organizing Swing components with Layout Managers

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

import ...

public class NumberGrid extends JFrame {

    private JButton[] btnNumbers = new JButton[10]; 3 usages
    private JButton btnDelete, btnReset; 3 usages
    private JTextField tfDisplay; 9 usages

    public NumberGrid() { 1 usage
        tfDisplay = new JTextField();
        tfDisplay.setPreferredSize(new Dimension( width: 200, height: 30));
        tfDisplay.setComponentOrientation(ComponentOrientation.RIGHT_TO_LEFT);

        JPanel panelButtons = new JPanel(new GridLayout( rows: 4, cols: 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( width: 200, height: 200);
        setVisible(true);
    }
}
```

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

void addButtons(JPanel panelButtons) { 1 usage
    ButtonListener btnListener = new ButtonListener();
    for (int i = 0; i <= 9; i++) {
        btnNumbers[i] = new JButton(String.valueOf(i));
        panelButtons.add(btnNumbers[i]);
        btnNumbers[i].addActionListener(btnListener);
    }
}
```

```

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

        btnReset = new JButton( text: "C");
        panelButtons.add(btnReset);
        btnReset.addActionListener(btnListener);
    }

    private class ButtonListener implements ActionListener { 2 usages
        @Override
        public void actionPerformed(ActionEvent e) {
            String button = e.getActionCommand();
            if (Character.isDigit(button.charAt(0))) {
                tfDisplay.setText(tfDisplay.getText() + button);
            } else if (button.equals("DEL")) {
                String deleteString = tfDisplay.getText();
                if (!deleteString.isEmpty()) {
                    tfDisplay.setText(deleteString.substring(0, deleteString.length() - 1));
                }
            } else {
                tfDisplay.setText("");
            }
        }
    }
}

public static String delLastCharacter(String str) { no usages
    return (str != null && !str.isEmpty()) ? str.substring(0, str.length() - 1) : null;
}
}

```

3. JavaFX API – GUIProject

```

package hust.soict.dsai.javaafx;

import ...

> public class Painter extends Application {
    @Override
    @ public void start(Stage stage) throws Exception {
        Parent root = FXMLLoader.load(getClass().getResource( name: "Painter.fxml"));
        Scene scene = new Scene(root);
        stage.setTitle("Painter");
        stage.setScene(scene);
        stage.show();
    }

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

```

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

import ...

public class PainterController { 1 usage
    @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(), v2: 4, Color.BLACK);
        drawingAreaPane.getChildren().add(newCircle);
    }
}
```

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.layout.Pane?>
<?import javafx.scene.layout.VBox?>
<?import javafx.scene.text.Font?>

<BorderPane fx:controller="hust.soict.dsai.javafx.PainterController" maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" style="-fx-background-color: #f0f0f0">
    <center>
        <BorderPane minHeight="-Infinity" minWidth="-Infinity" prefHeight="480.0" prefWidth="640.0" BorderPane.alignment="CENTER">
            <padding>
                <Insets bottom="8.0" left="8.0" right="8.0" top="8.0" />
            </padding>
            <center>
                <Pane fx:id="drawingAreaPane" onMouseDragged="#drawingAreaMouseDragged" prefHeight="200.0" prefWidth="200.0" style="-fx-background-color: #ffffff;" />
            </center>
        </BorderPane>
    </center>
    <left>
        <VBox fx:id="toolMenu" maxHeight="1.7976931348623157E308" spacing="8.0" BorderPane.alignment="CENTER">
            <BorderPane.margin>
                <Insets right="8.0" />
            </BorderPane.margin>
            <children>
                <Button maxWidth="1.7976931348623157E308" mnemonicParsing="false" onAction="#clearButtonPressed" text="Clear">
                    <font>
                        <Font size="13.0" />
                    </font>
                </Button>
            </children>
        </VBox>
    </left>
</BorderPane>
```

4. JavaFX API – AimsProject StoreScreen.java

```

package hust.soict.dsai.aims.screen;

> import ...

public class StoreScreen extends JFrame { 5 usages
    private final Store store; 5 usages
    private final Cart cart; 4 usages
    private JPanel centerPanel; 5 usages

    public StoreScreen(Store store, Cart cart) { 2 usages
        this.store = store;
        this.cart = cart;

        setTitle("Store");
        setSize( width: 1024, height: 768);
        setDefaultCloseOperation(EXIT_ON_CLOSE);

        Container cp = getContentPane();
        cp.setLayout(new BorderLayout());

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

        setVisible(true);
    }

    private JPanel createNorth() { 1 usage
        JPanel north = new JPanel();
        north.setLayout(new BoxLayout(north, BoxLayout.Y_AXIS));

        north.add(createMenuBar());
        north.add(createHeader());

        return north;
    }

    private JMenuBar createMenuBar() { 1 usage
        JMenu menu = new JMenu( s: "Options");

        JMenu smUpdateStore = new JMenu( s: "Update Store");
        JMenuItem addBook = new JMenuItem( text: "Add Book");
        addBook.addActionListener( ActionEvent e -> new AddBookToStoreScreen(store));
        smUpdateStore.add(addBook);
    }

```



```

private JMenuBar createMenuBar() { 1 usage
    JMenu menu = new JMenu( s: "Options");

    JMenu smUpdateStore = new JMenu( s: "Update Store");
    JMenuItem addBook = new JMenuItem( text: "Add Book");
    addBook.addActionListener( ActionEvent e -> new AddBookToStoreScreen(store));
    smUpdateStore.add(addBook);

    JMenuItem addCD = new JMenuItem( text: "Add CD");
    addCD.addActionListener( ActionEvent e -> new AddCompactDiscToStoreScreen(store));
    smUpdateStore.add(addCD);

    JMenuItem addDVD = new JMenuItem( text: "Add DVD");
    addDVD.addActionListener( ActionEvent e -> new AddDigitalVideoDiscToStoreScreen(store));
    smUpdateStore.add(addDVD);

    menu.add(smUpdateStore);

    JMenuItem viewCart = new JMenuItem( text: "View Cart");
    viewCart.addActionListener( ActionEvent e -> new CartScreen(cart));
    menu.add(viewCart);

    JMenuBar menuBar = new JMenuBar();
    menuBar.add(menu);
    return menuBar;
}

private JPanel createHeader() { 1 usage
    JPanel header = new JPanel();
    header.setLayout(new BoxLayout(header, BoxLayout.X_AXIS));

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

    JButton cartButton = new JButton( text: "View Cart");
    cartButton.addActionListener( ActionEvent e -> new CartScreen(cart));

```

```

        cartButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                new CarScreen(cart);
            }
        });

        header.add(Box.createRigidArea(new Dimension( width: 10, height: 10)));
        header.add(title);
        header.add(Box.createHorizontalGlue());
        header.add(cartButton);
        header.add(Box.createRigidArea(new Dimension( width: 10, height: 10)));

        return header;
    }

    private JPanel createCenter() { 2 usages
        JPanel center = new JPanel();
        center.setLayout(new GridLayout( rows: 3, cols: 3, hgap: 2, vgap: 2));

        store.getItemsInStore().forEach( Media media -> {
            MediaStore cell = new MediaStore(media, cart);
            center.add(cell);
        });
        return center;
    }

    public void refreshCenter() { 3 usages
        // Remove old center panel
        getContentPane().remove(centerPanel);
        centerPanel = createCenter();
        getContentPane().add(centerPanel, BorderLayout.CENTER);
        validate();
        repaint();
    }
}

```

MediaStore.java


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

```
> import ...
```

```
public class MediaStore extends JPanel { 2 usages
```

```
    private final Media media; 5 usages
```

```
    private final Cart cart; 2 usages
```

```
    public MediaStore(Media media, Cart cart) { 1 usage
```

```
        this.media = media;
```

```
        this.cart = cart;
```

```
        this.setLayout(new BoxLayout(target: this, BoxLayout.Y_AXIS));
```

```
        // Title Label
```

```
        JLabel title = new JLabel(media.getTitle());
```

```
        title.setFont(new Font(title.getFont().getName(), Font.PLAIN, size: 20));
```

```
        title.setAlignmentX(CENTER_ALIGNMENT);
```

```
        // Cost Label
```

```
        JLabel cost = new JLabel(String.format("%.2f $", media.getCost()));
```

```
        cost.setAlignmentX(CENTER_ALIGNMENT);
```

```
        // Button Panel
```

```
        JPanel container = new JPanel();
```

```
        container.setLayout(new FlowLayout(FlowLayout.CENTER));
```

```
        JButton addToCartButton = new JButton(text: "Add to cart");
```

```
        addToCartButton.addActionListener(new AddToCartListener());
```

```
        container.add(addToCartButton);
```

```
        if (media instanceof Playable) {
```

```
            JButton playButton = new JButton(text: "Play");
```

```
            playButton.addActionListener(new PlayMediaListener());
```

```
            container.add(playButton);
```

```
        }
```

```
        // Add components to the panel
```

```
        this.add(Box.createVerticalGlue());
```

```
        this.add(title);
```

```
        this.add(cost);
```

```
        this.add(Box.createVerticalGlue());
```

```
        this.add(container);
```

```

        // Set border for the media panel
        Border border = BorderFactory.createLineBorder(Color.BLACK);
        this.setBorder(border);
    }

    // Listener for Add to Cart Button
    private class AddToCartListener implements ActionListener { 1 usage
        @Override
        public void actionPerformed(ActionEvent e) {
            cart.addMedia(media);
            JOptionPane.showMessageDialog( parentComponent: null, message: media.getTitle() + " added to cart.");
        }
    }

    // Listener for Play Button
    private class PlayMediaListener implements ActionListener { 1 usage
        @Override
        public void actionPerformed(ActionEvent e) {
            if (media instanceof Playable) {
                try {
                    ((Playable) media).play();
                } catch (PlayerException ex) {
                    throw new RuntimeException(ex);
                }
            }
        }
    }
}

```

CastScreenController.java

```

package hust.soict.dsai.aims.screen;

import ...

public class CartScreenController { 2 usages
    private final Cart cart; 6 usages

    @FXML 5 usages
    private TableView<Media> tblMedia;

    @FXML 1 usage
    private TableColumn<Media, String> colMediaTitle;

    @FXML 1 usage
    private TableColumn<Media, String> colMediaCategory;

    @FXML 1 usage
    private TableColumn<Media, Double> colMediaCost;

    @FXML 1 usage
    private Label lblTotalCost;

    @FXML 1 usage
    private TextField txtFilter;
}

```

```

@FXML 1 usage
private RadioButton radioBtnFilterId;

@FXML 1 usage
private RadioButton radioBtnFilterTitle;

@FXML 4 usages
private Button btnRemove;

@FXML 4 usages
private Button btnPlay;

private FilteredList<Media> filteredList; 3 usages

public CartScreenController(Cart cart) { this.cart = cart; }

@FXML no usages
private void initialize() {
    colMediaTitle.setCellValueFactory(new PropertyValueFactory<>("title"));
    colMediaCategory.setCellValueFactory(new PropertyValueFactory<>("category"));
    colMediaCost.setCellValueFactory(new PropertyValueFactory<>("cost"));

    filteredList = new FilteredList<>(cart.getItemsOrdered(), Media media -> true);
    tblMedia.setItems(filteredList);
    updateTotalCost();
    // Listen for cart changes to update total cost automatically
    cart.getItemsOrdered().addListener((java.util.ListChangeListener<Media> c) -> {
        tblMedia.refresh();
    });

    txtFilter.textProperty().addListener((ObservableValue<String> observable, String oldValue, String newValue) -> showFilteredMedia(n

    tblMedia.getSelectionModel().selectedItemProperty().addListener((ObservableValue<Media> observable, Media oldValue, Media newVa
        if (newValue != null) {
            updateButtonBar(newValue);
        } else {
            btnRemove.setVisible(false);
            btnPlay.setVisible(false);
        }
    });
}

```

```

        btnRemove.setVisible(false);
        btnPlay.setVisible(false);
    }

    private void applyFilter(String filter) { 1 usage
        filteredList.setPredicate(Media media -> {
            if (filter == null || filter.isEmpty()) {
                return true;
            }

            String lowerCaseFilter = filter.toLowerCase();
            if (radioBtnFilterId.isSelected()) {
                return String.valueOf(media.getId()).contains(lowerCaseFilter);
            } else if (radioBtnFilterTitle.isSelected()) {
                return media.getTitle().toLowerCase().contains(lowerCaseFilter);
            }
            return false;
        });
    }
}

```

```

private void updateTotalCost() { lblTotalCost.setText(String.format("%.2f", cart.totalCost())); }

private void showFilteredMedia(String filter) { applyFilter(filter); }

private void updateButtonBar(Media media) { 1 usage
    btnRemove.setVisible(true);
    btnPlay.setVisible(media instanceof hust.soict.dsai.aims.media.Playable);
}

@FXML no usages
private void btnRemovePressed(ActionEvent event) {
    Media selectedMedia = tblMedia.getSelectionModel().getSelectedItem();
    if (selectedMedia != null) {
        cart.removeMedia(selectedMedia);
        updateTotalCost();
        btnRemove.setVisible(false);
        btnPlay.setVisible(false);
        System.out.println("Removed: " + selectedMedia.getTitle());
    } else {
        System.out.println("No item selected for removal.");
    }
}

@FXML no usages
private void btnPlayPressed(ActionEvent event) throws PlayerException {
    Media selectedMedia = tblMedia.getSelectionModel().getSelectedItem();
    if (selectedMedia instanceof hust.soict.dsai.aims.media.Playable playable) {
        playable.play();
        Alert alert = new Alert(Alert.AlertType.INFORMATION);
        alert.setTitle("Playing Media");
        alert.setHeaderText(null);
        alert.setContentText("Playing: " + selectedMedia.getTitle());
        alert.showAndWait();
    }
}

@FXML no usages
private void btnPlaceOrderPressed(ActionEvent event) {
    cart.getItemsOrdered().clear();
    updateTotalCost();
    System.out.println("Order placed successfully!");
}

```

```

@FXML no usages
private void menuViewStoreAction(ActionEvent event) {
    if (Aims.getStoreScreenInstance() != null) {
        Aims.getStoreScreenInstance().setExtendedState(java.awt.Frame.NORMAL);
        Aims.getStoreScreenInstance().toFront();
        Aims.getStoreScreenInstance().requestFocus();
    } else {
        // Fallback if instance is null
        new StoreScreen(Aims.getStore(), Aims.getCart());
    }
}

```

CartScreen.java

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

import ...

public class CartScreen extends JFrame { 2 usages
    private final Cart cart; 1 usage

    public CartScreen(Cart cart) { 2 usages
        super();
        this.cart = cart;

        // Create a JavaFX panel
        JFXPanel fxPanel = new JFXPanel();
        this.add(fxPanel);

        // Set up the JFrame
        this.setTitle("Cart");
        this.setSize( width: 1024, height: 768);
        this.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        this.setVisible(true);

        // Load JavaFX content into the JFXPanel
        Platform.runLater(() -> {
            try {
                FXMLLoader loader = new FXMLLoader(getClass().getResource( name: "Cart.fxml"));
                CartScreenController controller = new CartScreenController(cart);
                loader.setController(controller);
                Parent root = loader.load();
                fxPanel.setScene(new Scene(root));
            } catch (IOException e) {
                e.printStackTrace();
                JOptionPane.showMessageDialog( parentComponent: this, message: "Unable to load Cart Screen", title: "Error", JOptionPane.ERROR_MESSA
            }
        });
    }
}
```

AddItemToStoreScreen.java

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

import ...

public abstract class AddItemToStoreScreen extends JFrame { 3 usages 3 inheritors 1 Github1byNz
    protected Store store; 5 usages
    protected JTextField titleField; 6 usages
    protected JTextField categoryField; 6 usages
    protected JTextField costField; 7 usages

    public AddItemToStoreScreen(Store store) { 3 usages 1 Github1byNz
        this.store = store;

        // Setup frame
        setTitle("Add Item to Store");
        setSize( width: 400, height: 300);
        setDefaultCloseOperation(DISPOSE_ON_CLOSE);
        setLayout(new BorderLayout());

        // Input fields panel
        JPanel inputPanel = new JPanel(new GridLayout( rows: 3, cols: 2, hgap: 10, vgap: 10));
        inputPanel.add(new JLabel( text: "Title:"));
        titleField = new JTextField();
        inputPanel.add(titleField);
    }
}
```

```

inputPanel.add(new JLabel( text: "Category:"));
categoryField = new JTextField();
inputPanel.add(categoryField);

inputPanel.add(new JLabel( text: "Cost:"));
costField = new JTextField();
inputPanel.add(costField);

add(inputPanel, BorderLayout.CENTER);

// Buttons panel
JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.RIGHT));
JButton addButton = new JButton( text: "Add");
addButton.addActionListener(new AddButtonListener());
buttonPanel.add(addButton);

JButton cancelButton = new JButton( text: "Cancel");
cancelButton.addActionListener( ActionEvent e -> this.dispose());
buttonPanel.add(cancelButton);

add(buttonPanel, BorderLayout.SOUTH);

setVisible(true);
}

```

```

// Validate common inputs
protected boolean validateInputs() { 1 usage  ⓘ GithubbyNz
    if (titleField.getText().trim().isEmpty() ||
        categoryField.getText().trim().isEmpty() ||
        costField.getText().trim().isEmpty()) {
        JOptionPane.showMessageDialog( parentComponent: this, message: "All fields must be filled out.", title: "Validation Error", JOptionPane.
        return false;
    }

    try {
        double cost = Double.parseDouble(costField.getText());
        if (cost <= 0) {
            JOptionPane.showMessageDialog( parentComponent: this, message: "Cost must be greater than zero.", title: "Validation Error", JOptionPane.
            return false;
        }
    } catch (NumberFormatException e) {
        JOptionPane.showMessageDialog( parentComponent: this, message: "Cost must be a valid number.", title: "Validation Error", JOptionPane.ER
        return false;
    }

    return true;
}

```

```

// Abstract method for subclasses to define specific add behavior
protected abstract void handleAddAction(); 1 usage 3 implementations ⓘ GithubbyNz

// Listener for the Add button
private class AddButtonListener implements ActionListener { 1 usage ⓘ GithubbyNz
    @Override ⓘ GithubbyNz
    public void actionPerformed(ActionEvent e) {
        if (validateInputs()) {
            handleAddAction(); // Delegate to subclass implementation
        }
    }
}

```


AddDigitalVideoDiscToStoreScreen.java

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

D:\Lab05\AimsProject\src\main\java\hust\soict\dsai\aims\screen\MediaStore.java

import ...

public class AddDigitalVideoDiscToStoreScreen extends AddItemToStoreScreen { 1 usage 1 Github1byNz
    private final JTextField directorField; 3 usages
    private final JTextField lengthField; 3 usages

    public AddDigitalVideoDiscToStoreScreen(Store store) { 1 usage 1 Github1byNz
        super(store);

        setSize( width: 500, height: 400);
        setLocationRelativeTo(null);

        JPanel dvdPanel = new JPanel(new GridLayout( rows: 2, cols: 2, hgap: 10, vgap: 10));
        dvdPanel.setBorder(BorderFactory.createEmptyBorder( top: 10, left: 10, bottom: 10, right: 10));

        dvdPanel.add(new JLabel( text: "Director:"));
        directorField = new JTextField();
        dvdPanel.add(directorField);

        dvdPanel.add(new JLabel( text: "Length (minutes):"));
        lengthField = new JTextField();
        dvdPanel.add(lengthField);

        add(dvdPanel, BorderLayout.SOUTH);
    }
}
```

```
@Override 1 usage 1 Github1byNz
protected void handleAddAction() {
    try {
        String title = titleField.getText().trim();
        String category = categoryField.getText().trim();
        double cost = Double.parseDouble(costField.getText().trim());
        String director = directorField.getText().trim();
        int length = Integer.parseInt(lengthField.getText().trim());

        DigitalVideoDisc dvd = new DigitalVideoDisc(title, category, director, length, cost);
        store.addMedia(dvd);
        JOptionPane.showMessageDialog( parentComponent: this, message: "DVD added successfully!");
        Aims.getStoreScreenInstance().refreshCenter();
        this.dispose();
    } catch (NumberFormatException ex) {
        JOptionPane.showMessageDialog( parentComponent: this, message: "Invalid input for cost or length. Please enter valid numbers.", title: "
    } catch (Exception ex) {
        JOptionPane.showMessageDialog( parentComponent: this, message: "An error occurred. Please check your inputs and try again.", title: "Err
    }
}
```

AddCompactDiscToStoreScreen.java

```

package hust.soict.dsai.aims.screen;

import ...

public class AddCompactDiscToStoreScreen extends AddItemToStoreScreen { 1 usage 1 Github1byNz
    private final JTextField artistField; 3 usages
    private final JTextField directorField; 3 usages
    private final JTextArea tracksArea; 3 usages

    public AddCompactDiscToStoreScreen(Store store) { 1 usage 1 Github1byNz
        super(store);

        setSize( width: 500, height: 400);
        setLocationRelativeTo(null);

        JPanel cdPanel = new JPanel(new GridLayout( rows: 3, cols: 2, hgap: 10, vgap: 10));
        cdPanel.setBorder(BorderFactory.createEmptyBorder( top: 10, left: 10, bottom: 10, right: 10));

        cdPanel.add(new JLabel( text: "Artist:"));
        artistField = new JTextField();
        cdPanel.add(artistField);

        cdPanel.add(new JLabel( text: "Director:"));
        directorField = new JTextField();
        cdPanel.add(directorField);

        cdPanel.add(new JLabel( text: "Tracks (comma-separated):"));
        tracksArea = new JTextArea();
        cdPanel.add(tracksArea);

        add(cdPanel, BorderLayout.SOUTH);
    }

    @Override 1 usage 1 Github1byNz
    protected void handleAddAction() {
        try {
            String title = titleField.getText().trim();
            String category = categoryField.getText().trim();
            double cost = Double.parseDouble(costField.getText().trim());
            String artist = artistField.getText().trim();
            String director = directorField.getText().trim();
            String[] trackNames = tracksArea.getText().split( regex: ",");

```

```

            CompactDisc cd = new CompactDisc(title, category, artist, director, cost);

            Arrays.stream(trackNames)
                .map(String::trim)
                .filter( String track -> !track.isEmpty())
                .forEach( String track -> cd.addTrack(new Track(track, length: 0));

            store.addMedia(cd);
            JOptionPane.showMessageDialog( parentComponent: this, message: "CD added successfully!");
            Aims.getStoreScreenInstance().refreshCenter();
            this.dispose();
        } catch (Exception ex) {
            JOptionPane.showMessageDialog( parentComponent: this, message: "Invalid input. Please try again.", title: "Error", JOptionPane.ERROR_MESSAGE);
        }
    }
}

```

AddBookToStoreScreen.java

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

import ...

public class AddBookToStoreScreen extends AddItemToStoreScreen {
    private final JTextArea authorsArea;

    public AddBookToStoreScreen(Store store) {
        super(store);

        // Set a larger size and center on screen
        setSize( width: 500, height: 400);
        setLocationRelativeTo(null);

        JPanel bookPanel = new JPanel(new GridLayout( rows: 1, cols: 2, hgap: 10, vgap: 10));
        bookPanel.setBorder(BorderFactory.createEmptyBorder( top: 10, left: 10, bottom: 10, right: 10));

        bookPanel.add(new JLabel( text: "Authors (comma-separated):"));
        authorsArea = new JTextArea();
        bookPanel.add(authorsArea);

        add(bookPanel, BorderLayout.SOUTH);
    }

    @Override
    protected void handleAddAction() {
        try {
            String title = titleField.getText().trim();
            String category = categoryField.getText().trim();
            double cost = Double.parseDouble(costField.getText().trim());
            String[] authors = authorsArea.getText().split( regex: "," );

            Book book = new Book(store.getNextId(), title, category, cost);
            for (String author : authors) {
                if (!author.trim().isEmpty()) {
                    book.addAuthor(author.trim());
                }
            }

            store.addMedia(book);
            JOptionPane.showMessageDialog( parentComponent: this, message: "Book added successfully!");
            Aims.getStoreScreenInstance().refreshCenter();
            this.dispose();
        } catch (Exception ex) {
            store.addMedia(book);
            JOptionPane.showMessageDialog( parentComponent: this, message: "Book added successfully!");
            Aims.getStoreScreenInstance().refreshCenter();
            this.dispose();
        } catch (Exception ex) {
            JOptionPane.showMessageDialog( parentComponent: this, message: "Invalid input. Please try again.", title: "Error", JOptionPane.ERROR_MES
        }
    }
}
```

TestMediaEquals.java

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

> import ...

> public class TestMediaEquals { new *
    /unlikely-arg-type/ new *
> ~
    public static void main(String[] args) throws Exception {
        Media book1 = new Book( id: 1, title: "Harry Potter", category: "Fantasy", cost: 19.99f);
        Media book2 = new Book( id: 2, title: "Harry Potter", category: "Adventure", cost: 25.99f);
        ⚡ Media book3 = new Book( id: 3, title: "Lord of the Rings", category: "Fantasy", cost: 29.99f);

        // Kiểm tra equals()
        System.out.println("book1 equals book2? " + book1.equals(book2)); // true
        System.out.println("book1 equals book3? " + book1.equals(book3)); // false

        // Kiểm tra null và khác kiểu
        System.out.println("book1 equals null? " + book1.equals(null)); // false
        System.out.println("book1 equals String? " + book1.equals("Harry Potter")); // false
    }
}
```

Exception

