**Lab 05: GUI Programming**

**Bùi Ý Nhi\_20225657**

Table of Contents

1. [New written code 2](#_bookmark0)
   1. [Swing Component 2](#_bookmark1)
   2. [Tổ chức Swing components with Layout Managers 4](#_bookmark2)
   3. [Tạo GUI cho AIMS với Swing 5](#_bookmark3)
   4. [API JavaFX 9](#_bookmark4)
   5. [View Cart Screen với Scene Builder 12](#_bookmark5)
   6. [Tích hợp JavaFX vào Swing – Lớp JFXPanel 12](#_bookmark6)
   7. [Hiển thị các mục trong Store – JavaFX’s data-driven UI 13](#_bookmark7)
   8. [Cập nhật các nút trong TableView – ChangeListener 14](#_bookmark8)
   9. [Xóa 1 Media – nút btnRemove 15](#_bookmark9)
   10. [Lọc các mục trong giỏ hàng 15](#_bookmark10)
   11. [Hoàn thành ứng dụng GUI của Aims 16](#_bookmark11)
   12. [Kiểm tra source code trước đó để xử lý Exception 24](#_bookmark12)
   13. [Tạo một lớp kế thừa từ Exception 26](#_bookmark13)
   14. [Cập nhật lớp Aims 28](#_bookmark14)
   15. [Thay đổi method equals() của lớp Media 28](#_bookmark15)
2. [UML Diagram 29](#_bookmark16)
   1. [Use-Case Diagram 29](#_bookmark17)
   2. [Class Diagram (Final Lab05) 30](#_bookmark18)
   3. [Exception Hierachical 31](#_bookmark19)

# New written code

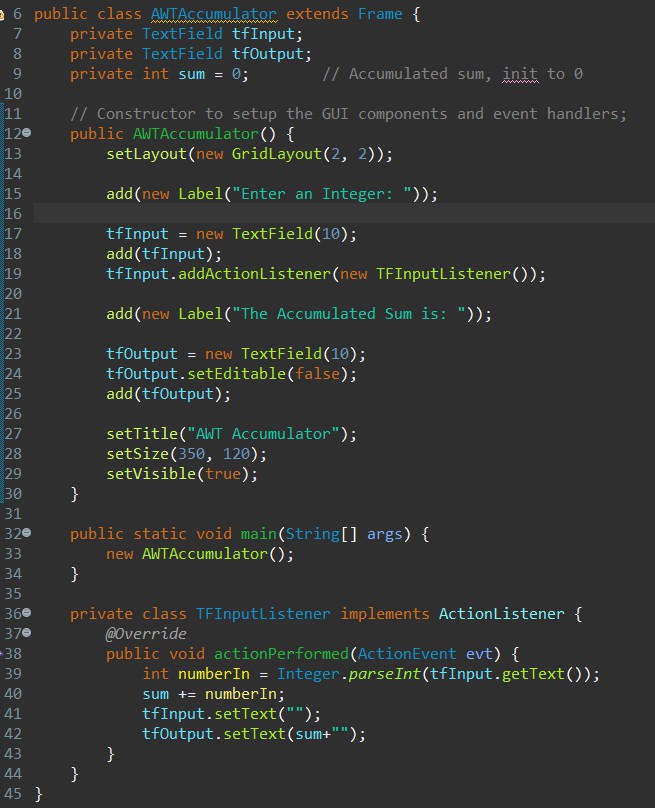
## Swing Component

#### Branch: topic/guiproject/swingcomponent\_awt

Tạo thư mục **GUIProject** đặt toàn bộ mã trong package **hust.soict.hedspi.swing**.

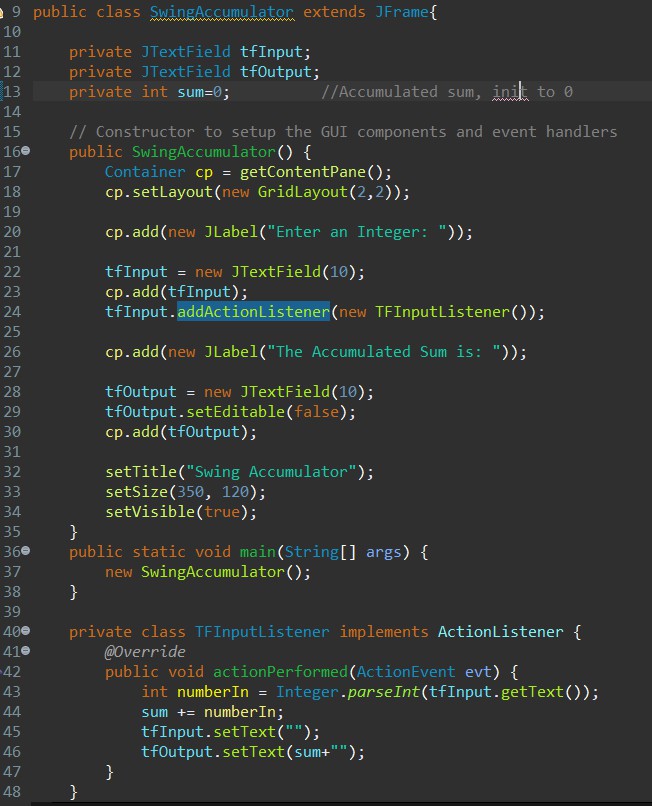
* 1. *AWTAccumulator*

Tạo **AWTAccumulator** tạo GUI để nhập vào số và tính tổng các số đã nhập.



* 1. *SwingAccumulator*

Tạo lớp **SwingAccumulator** với chức năng tương tự **AWTAccumulator**



* 1. *So sánh các thành phần Swing và AWT*

Lập trình với AWT và Swing khá giống nhau (bao gồm các thành phần/container, xử lý sự kiện). Tuy nhiên, có một số khác biệt cần lưu ý:

* + - Container cấp cao nhất trong Swing và AWT.
    - Tên lớp của các thành phần trong AWT và tên lớp tương ứng trong Swing.

## Tổ chức Swing components with Layout Managers

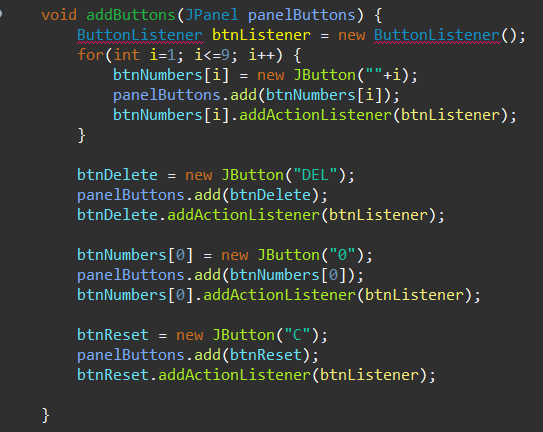
#### Branch: topic/guiproject/swingcomponent\_layoutmanagers

Sử dụng **Jpanel** như secondary-level container để tổ chức lại components.

* 1. *Tạo lớp NumberGrid*



* 1. *Thêm buttons*



* 1. *Hoàn thiện lớp bên trong ButtonListener*



## Tạo GUI cho AIMS với Swing

Đối với ứng dụng AIMS, triển khai ba màn hình:

* Màn hình **“View Store”** sử dụng **Swing**.
* Màn hình **“View Cart”** sử dụng **JavaFX.**
* Màn hình **“Update Store”** sử dụng **JavaFX**.

#### Branch: topic/aims-project/view-store-screen

* Với **View Store Screen** mã nguồn sẽ được đặt trong package

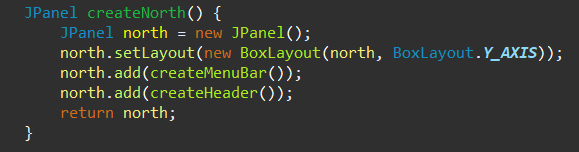
### hust.soict.hedspi.aims.screen

* Đối với màn hình View Store, chúng ta sẽ sử dụng **BorderLayout**:
* **Component NORTH** sẽ chứa **menu bar** và **header**.
* **Component CENTER** sẽ chứa một **panel** sử dụng **GridLayout**, mỗi ô trong đó là một mặt hàng của cửa hàng.
  1. *Tạo lớp StoreScreen*

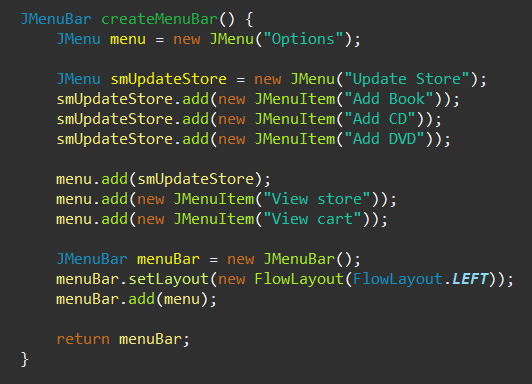
Đây là **View Store Screen**



* 1. *Component NORTH*



* Tạo **createMenuBar()**



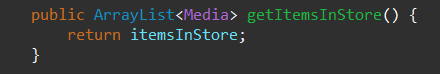
* Tạo **createHeader()**



* 1. *Componet CENTER*



- Thêm method **getItemsInStore** vào lớp **Store**



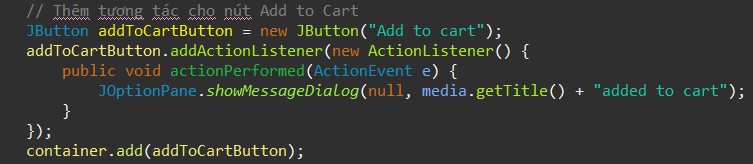
* 1. *Lớp MediaStore*



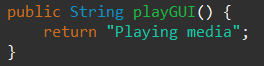
* 1. *Kết hợp tất cả*



* 1. *Thêm tương tác người dùng vào các nút trên Media*

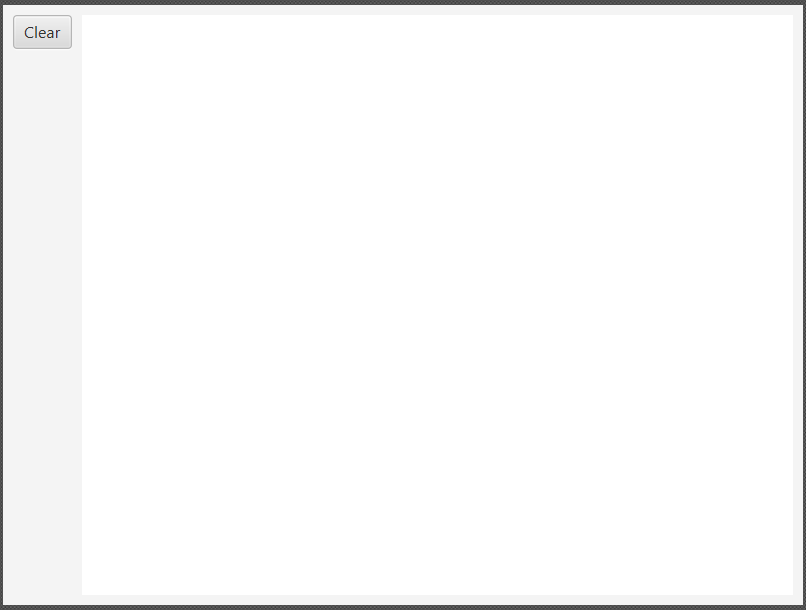


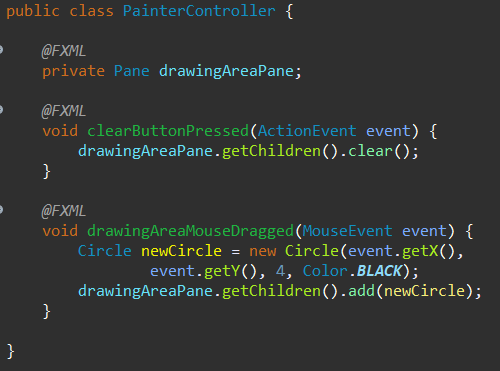
* + - Thêm method **playGUI()** cho lớp **Media**



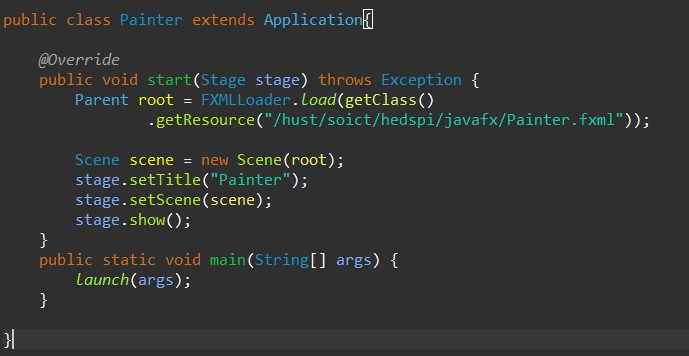
## API JavaFX

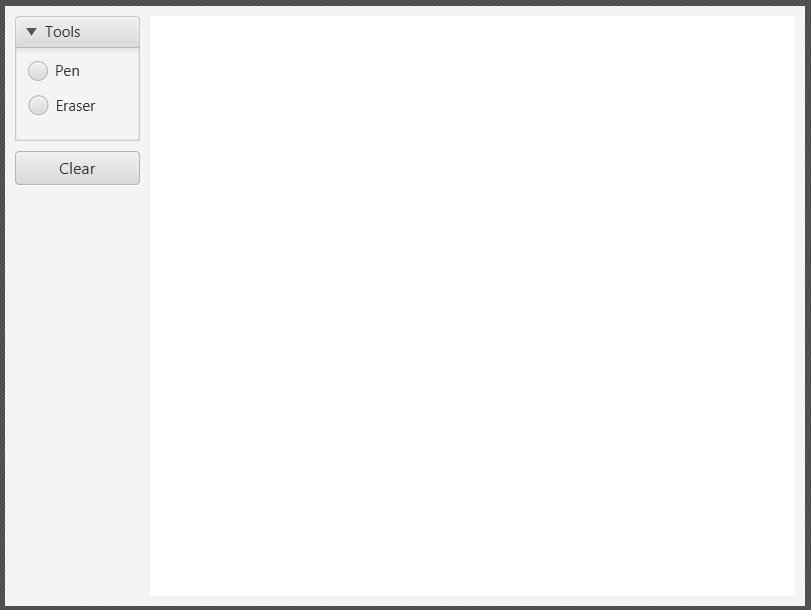
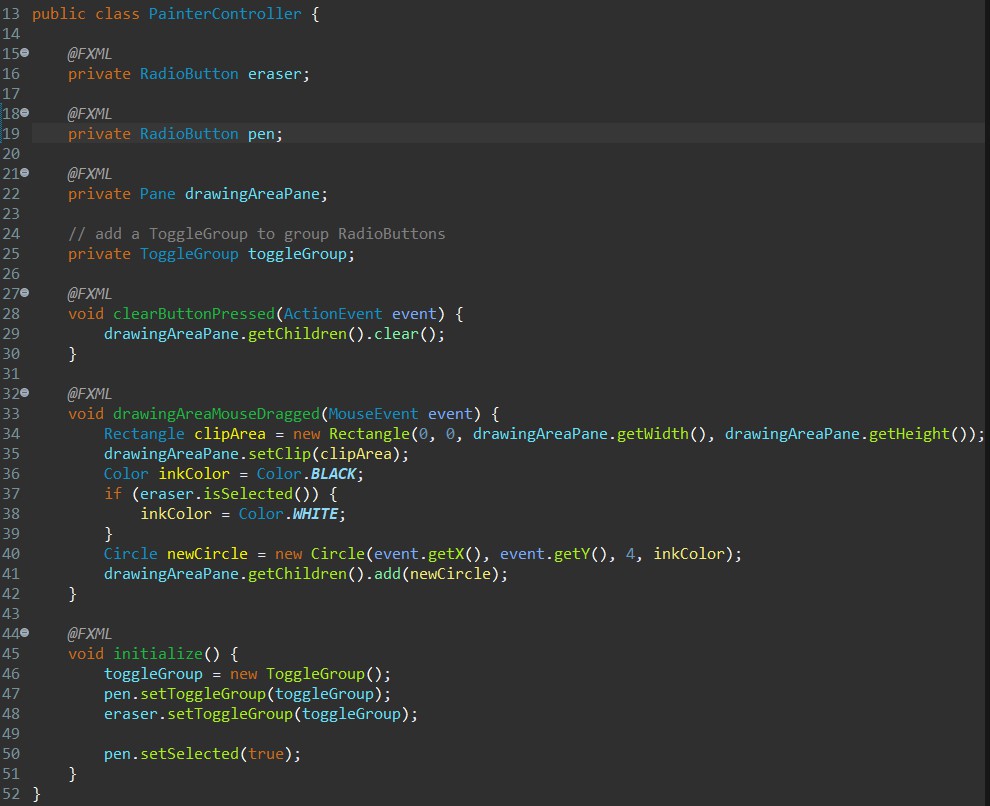
#### Branch: topic/guiproject/api-javafx

* 1. *Tạo GUI bằng SceneBuilder*
* Tại folder **GUIProject**
* Tạo file **Painter.fxml** trong **hust.soict.hedspi.javafx** và mở bằng SceneBuilder
  1. *Tạo lớp PainterController*



* 1. *Tạo lớp Painter để chạy Application*



* 1. *Thêm chức năng Eraser*
* Thêm **TitledPane** và 2 **RadioButtons** tương ứng với pen và eraser.
* Đặt thuộc tính **Toggle Group** cho cả 2 **RadioButtons** giống nhau để chỉ 1 trong số chúng được chọn tại 1 thời điểm
* Màn hình **Painter** sau khi thêm, file **Painter.fxml**
  + Chỉnh sửa lại **PainterController**

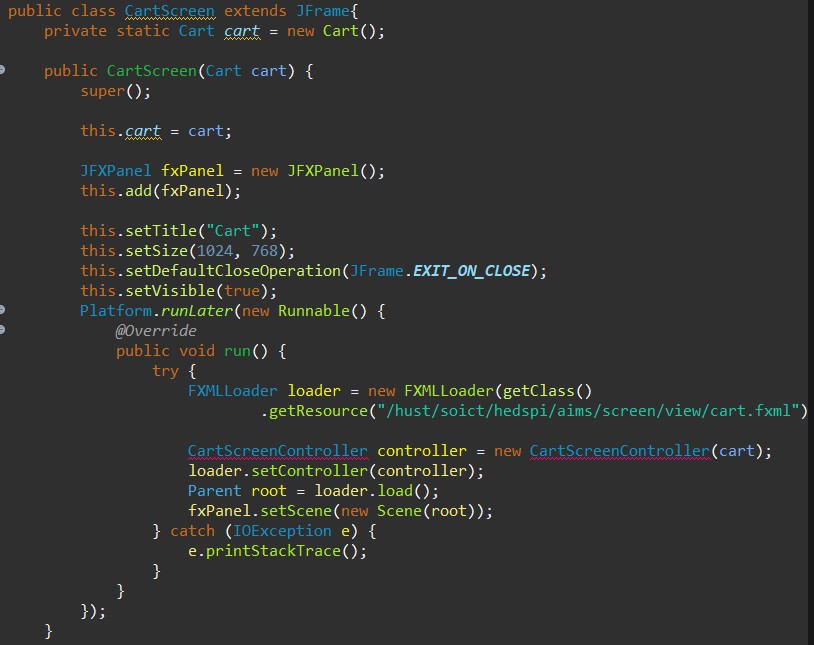
#### Branch: topic/aims-project/view-cart-screen (Từ 5-11)

## View Cart Screen với Scene Builder

Thực hành trong package **hust.soict.hedspi.aims.screen** với các bài tiếp theo.

### Tạo cart.fxml trong hust.soict.hedspi.aims.screen.view

## Tích hợp JavaFX vào Swing – Lớp JFXPanel

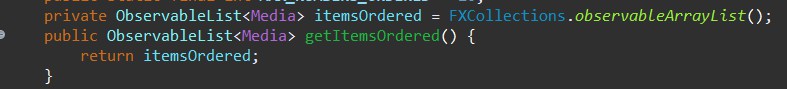
- Tạo lớp **CartScreen**

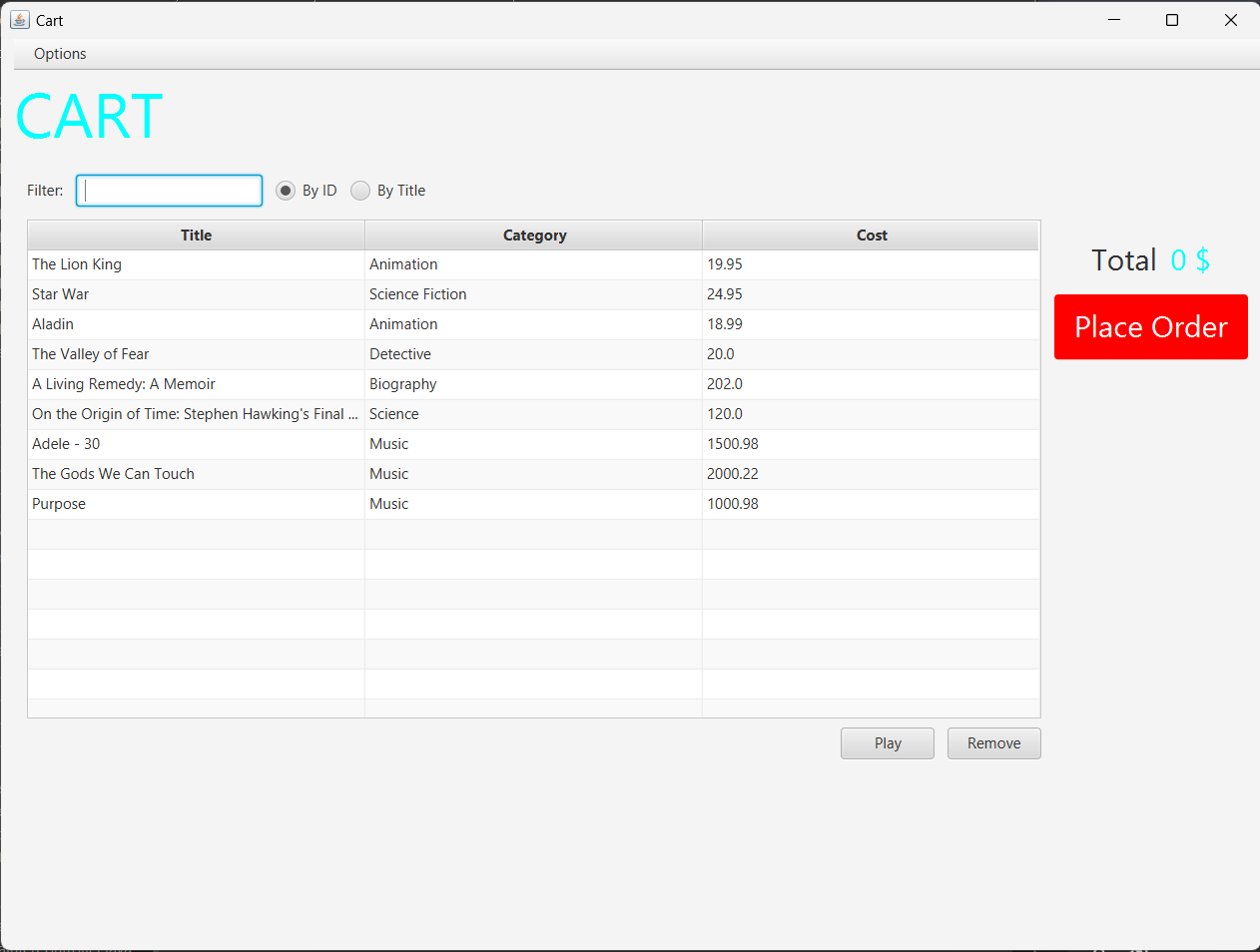
## Hiển thị các mục trong Store – JavaFX’s data-driven UI

### Tạo CartScreenController trong hust.soict.hedspi.aims.screen.controller



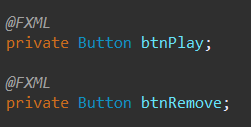
* + Thay đổi thuộc tính **itemsOrdered** từ **List<Media>** thành **ObservableList<Media>** và thêm method **getItemsOrdered()** trong lớp **Cart**



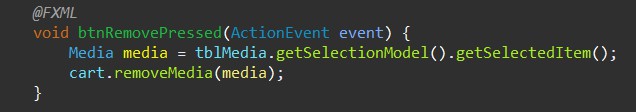
* + Kết quả màn hình **CartScreen**

## Cập nhật các nút trong TableView – ChangeListener

- Cập nhật nút **btnPlay** và **btnRemove** trong **CartScreenController**



## Xóa 1 Media – nút btnRemove

* Triển khai sự kiện cho nút Remove
  + Đồng thời thêm Action trong **Cart.fxml**

## Lọc các mục trong giỏ hàng

**TextField** nơi người dùng nhập chuỗi lọc: **tfFilter**

Hai nút radio để xác định tiêu chí lọc:

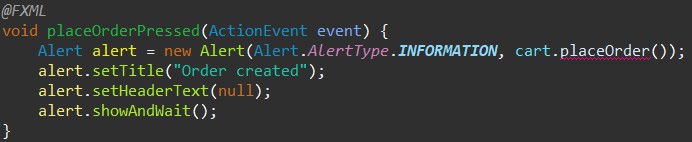
* Nút **“By ID”**: **radioBtnFilterId**

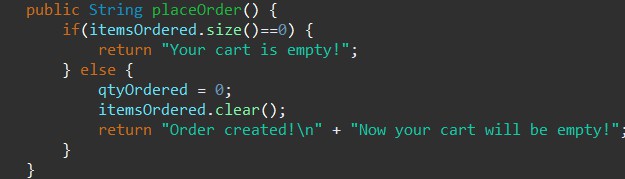
### Nút “By Title”: radioBtnFilterTitle

Thêm **ChangeListener** cho **tfFilter** trong **initialize()**



## Hoàn thành ứng dụng GUI của Aims

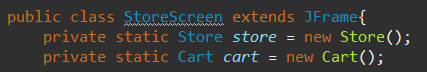
* 1. *Hoàn thành Cart Screen*
* Nút Place order 
  + Thêm method **placeOrder** trong lớp **Cart**

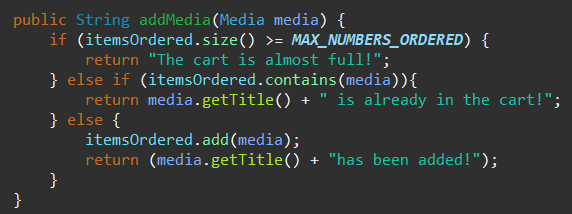


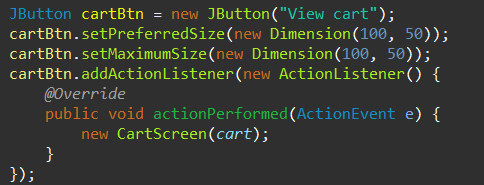
* Nút Play
* Cập nhật Cost vào thao tác xóa Media trong cart



#### Branch: topic/aims-project/update-store-screen

* Kết nối **StoreScreen** với **CartScreen**
  + Thêm new Cart vào lớp **StoreScreen**
  + Xóa **Cart, initsetup** ở lớp **CartScreen.java**
  + Mỗi lần ấn nút **Add to cart** thì add Media vào cart vừa tạo **(MediaStore.java)**
  + Sửa lại dạng trả về của method **addMedia()** trong **Cart.java**



* + Thêm action **View Cart** thì hiện ra **CartScreen** với những cart đã thêm
  1. *Update Store Screen*

Tạo 3 lớp

### AddDigitalVideoDiscToStoreScreen

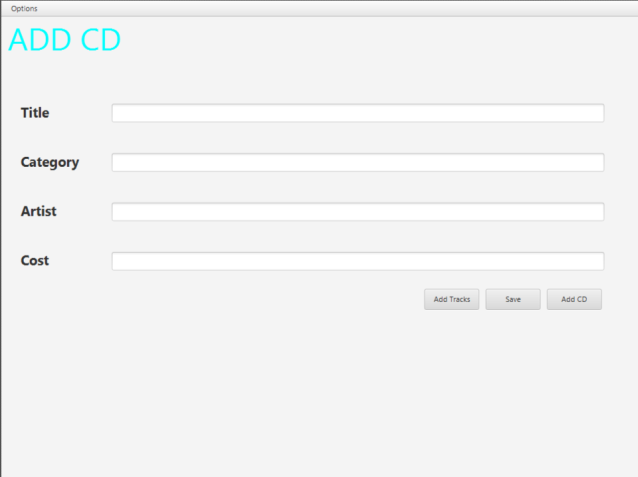
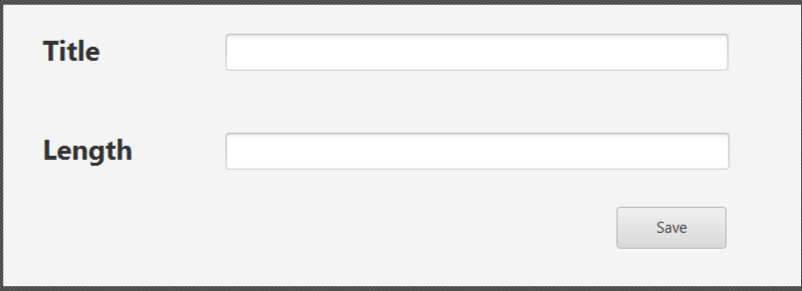
### AddBookToStoreScreen

### AddCompactDiscToStoreScreen

### AddTrackScreen

### Trong hust.soict.hedspi.aims.screen.view

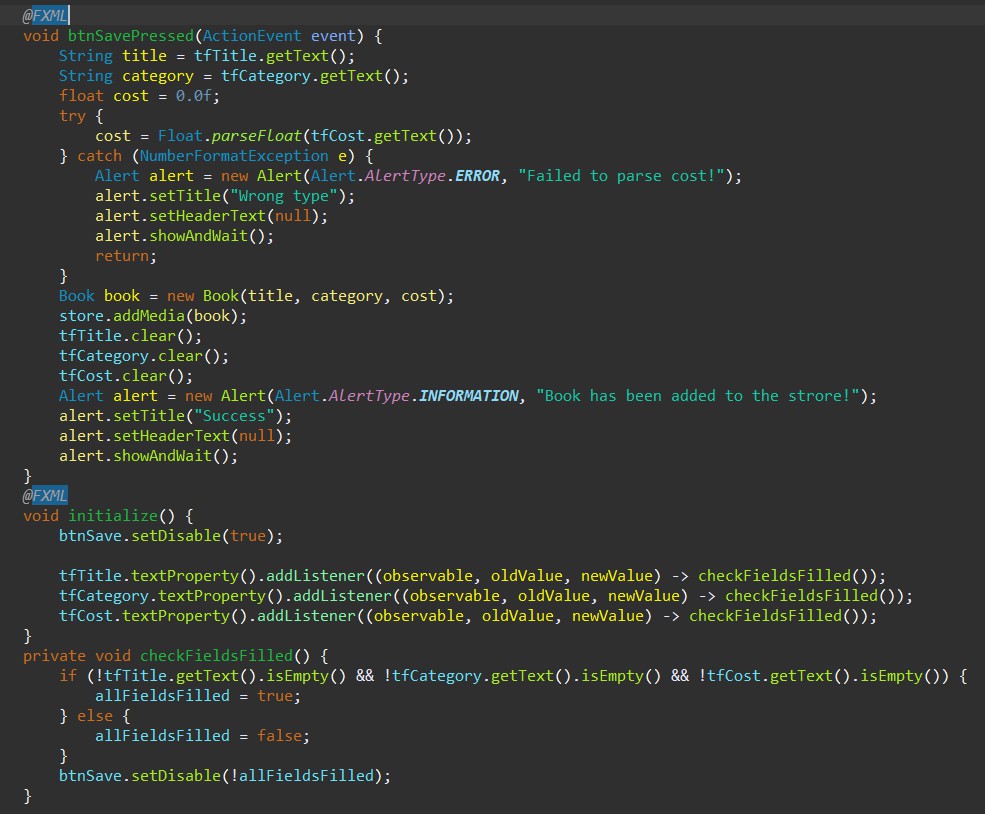
addBook.fxml addDVD.fxml

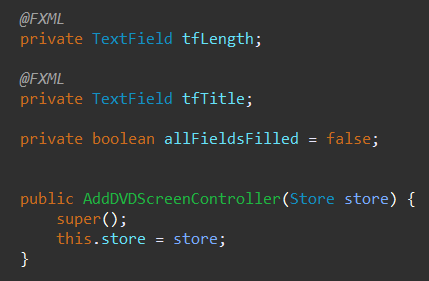
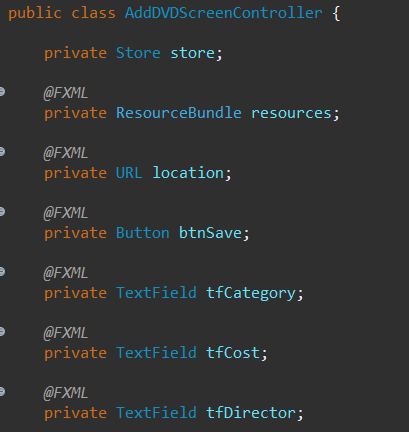
addCD.fxml addTracks.fxml

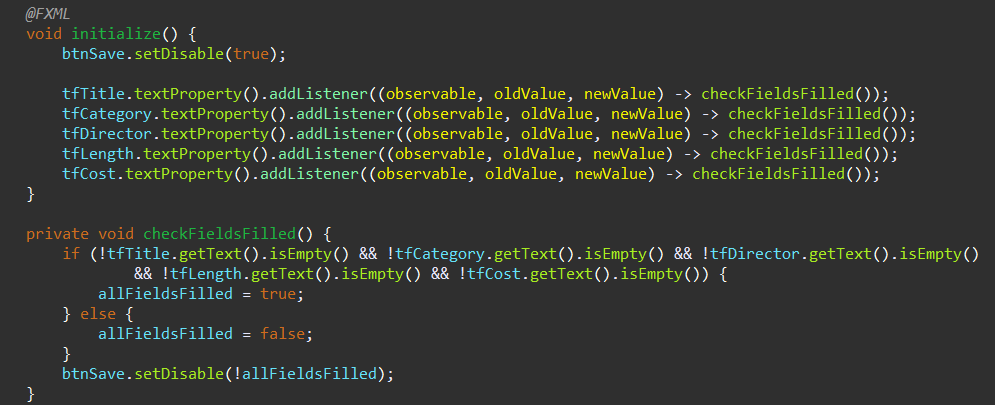
### Trong hust.soict.hedspi.aims.screen.controller

### AddBookScreenController.java

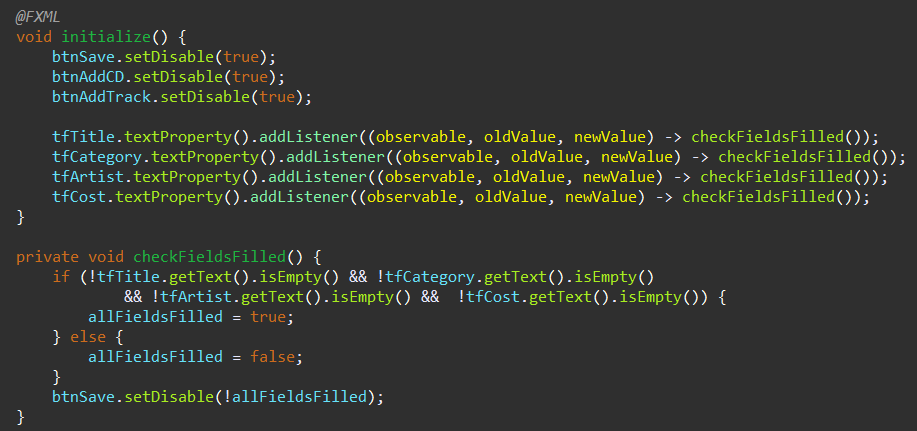


### AddDVDScreenController.java





### AddCDScreenController.java



### AddTrackScreenController.java



### Trong hust.soict.hedspi.aims.screen

Tạo ra 4 class

### AddBookToStoreScreen.java

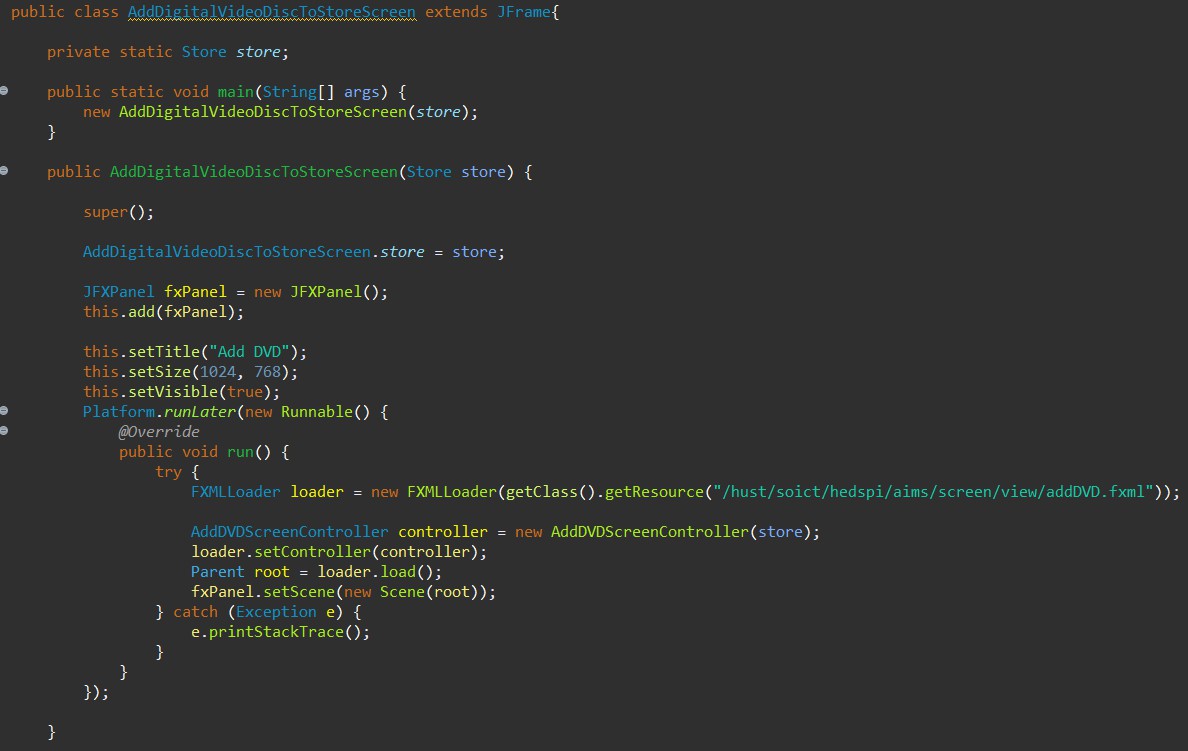
### AddDigitalVideoDiscToStoreScreen.java

### AddCompactDiscToStoreScreen.java

### AddTrack.java

4 Class này để có code giống nhau nên dưới đây chỉ minh họa 1 lớp là

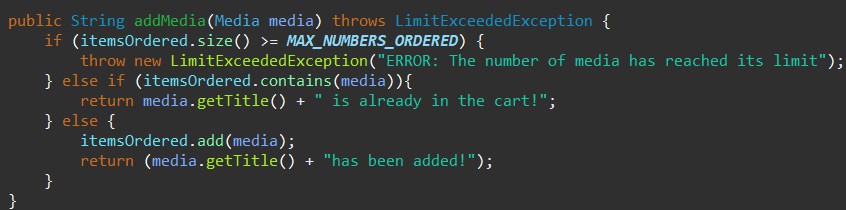
### AddDigitalVideoDiscToStoreScreen.java

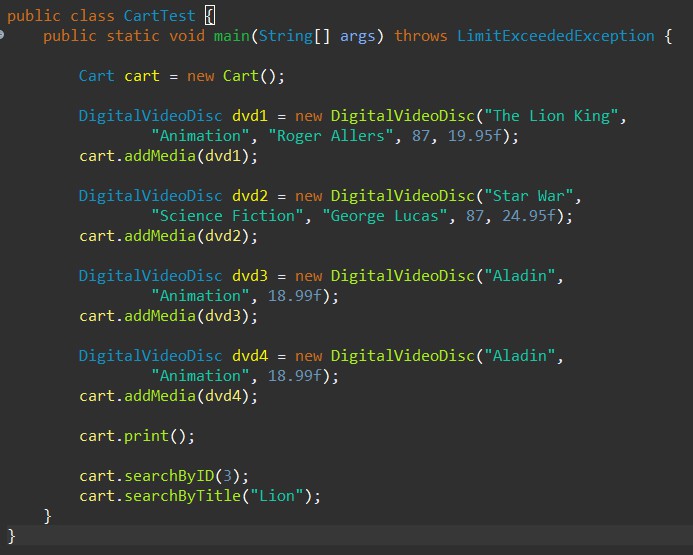


#### Branch: topic/aims-project/update-store-screen (12,13,14,15)

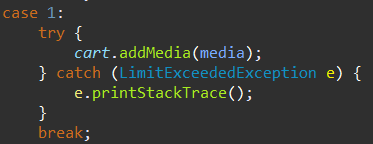
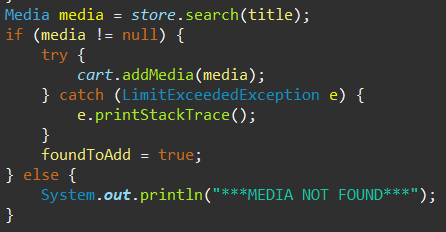
## Kiểm tra source code trước đó để xử lý Exception

* Xem xét tất cả các method, class trong lớp **AimsProject** để catch/handle tất cả những exception cần thiết.
* Throw Exception cho method **addMedia()** trong **Cart.java** và **testCart.java**





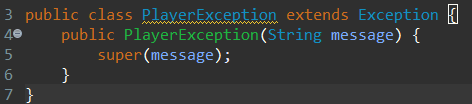
* Đồng thời catch Exception trong **MediaStore.java** và **Aims.java**



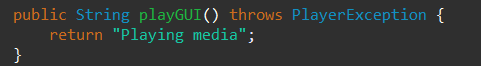
## Tạo một lớp kế thừa từ Exception

* 1. *Tạo 1 lớp mới có tên là PlayerException*

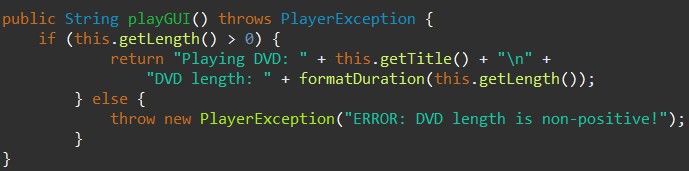
- Tạo 1 package **hust.soict.hedspi.aims.exception**, sau đó tạo lớp **PlayerException**



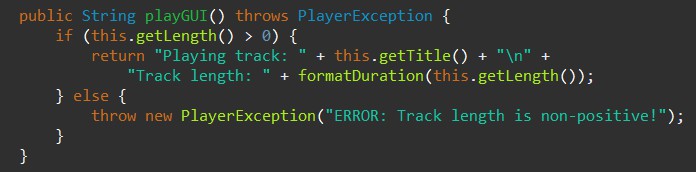
* 1. *Raise ngoại lệ PlayerException vào method play() (Bài này dùng method playGUI())*
* Thêm vào **Media**



* Thêm vào **DigitalVideoDisc**



* Thêm vào **Track**



* 1. *Cập nhật play() trong interface Playable*

- Không cập nhật vì ở đây dùng exception cho **playGUI()**

* 1. *Cập nhật play() trong lớp CompactDisc*

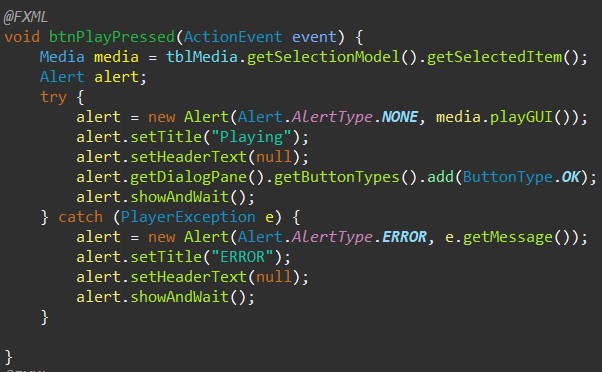


* Cập nhật các lớp khác

### MediaStore



### CartScreenController

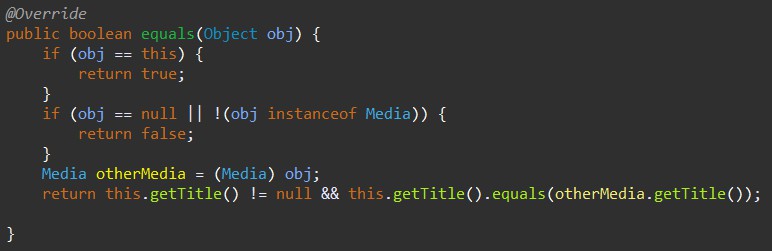


## Cập nhật lớp Aims

- Vì Exception được throw tại method **playGUI()** nên không ảnh hưởng lớp **Aims.**

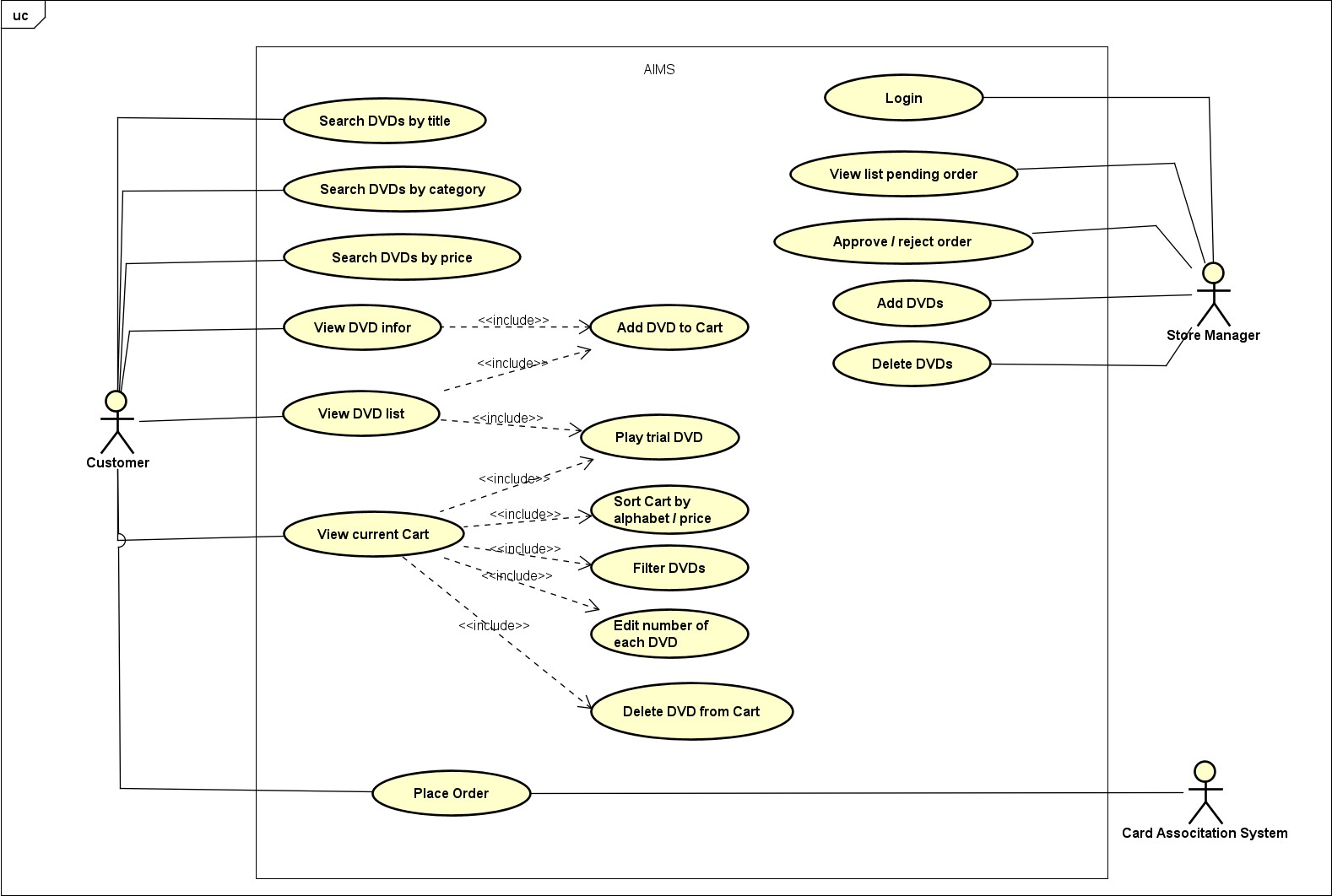
## Thay đổi method equals() của lớp Media

- Tránh gặp **NullPointerException** và **ClassCastException**

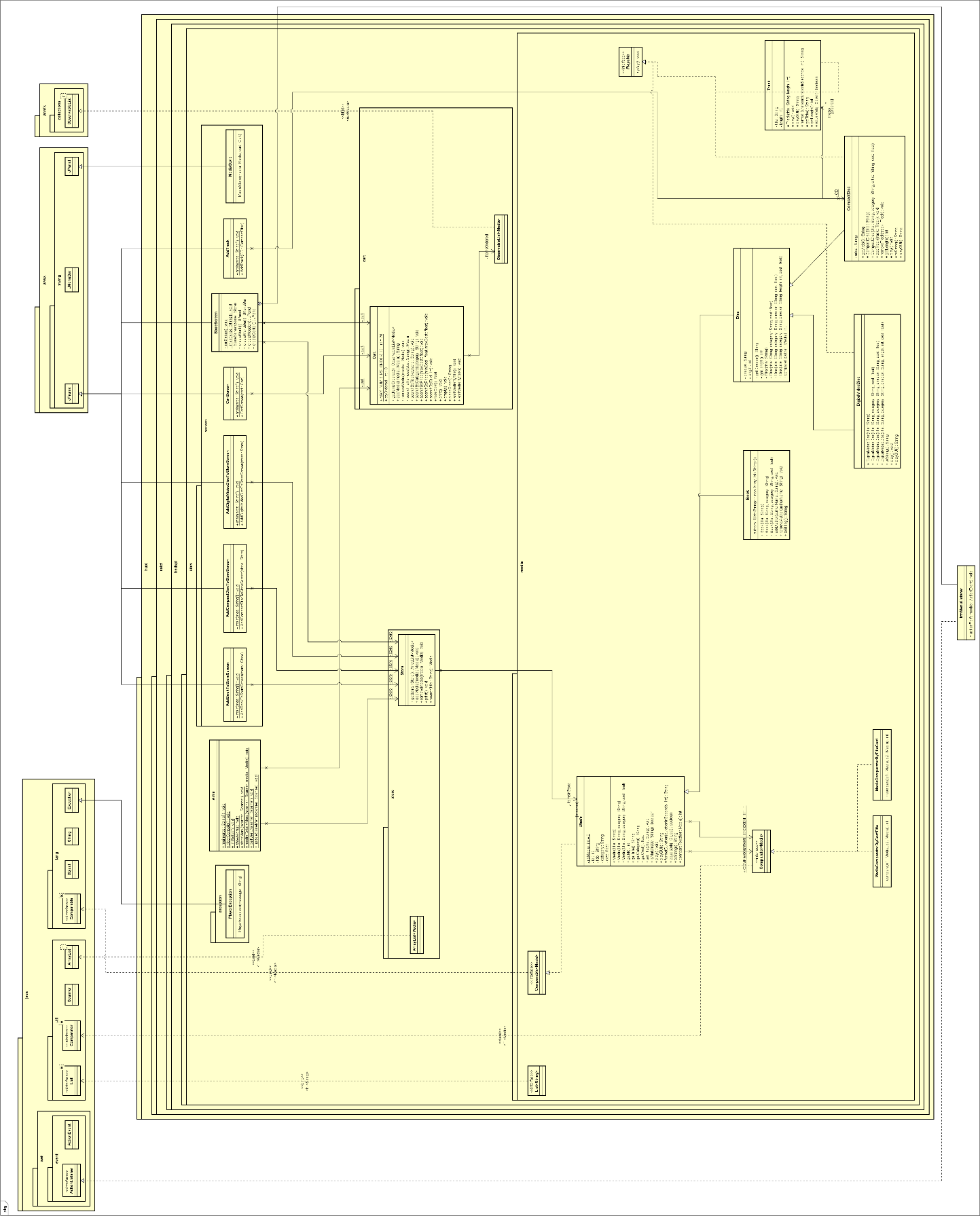


# UML Diagram

## Use-Case Diagram



## Class Diagram (Final Lab05)



## Exception Hierachical

