TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI

KHOA CÔNG NGHỆ THÔNG TIN

---------------o0o---------------



**BÀI TẬP LỚN MÔN HỌC**

**CÔNG NGHỆ JAVA**

Đề tài : Lập trình trò chơi chủ đề đi cảnh bằng Java  
Giảng viên: Đào Thị Lệ Thuỷ

Sinh viên thực hiện: Lê Đăng Sơn

Mã sinh viên: 221230972

*Hà Nội, tháng 4 năm 2024*

**LỜI CẢM ƠN**

Xin cảm ơn cô Đào Thị Lệ Thuỷ đã giảng giải về những kiến thức ngôn ngữ lập trình Java để hiện thực hoá ý tưởng của đề tài. Nhờ có cô, việc thực hiện đề tài có nền tảng vững chắc và kiến thức nền cho những nội dung mới trong quá trình nghiên cứu thực hiện. Những nội dung cơ bản trong ngôn ngữ lập trình Java mà cô giảng dạy là nền tảng quan trọng nhất trong quá trình thực hiện đề tài này.Cùng với sự kết hợp với các nguồn kiến thức khác, đề tài đã được hiện thực hoá một phần. Dù chưa hoàn chỉnh về mặt nội dung, những cơ chế và chức năng quan trọng đã được tích hợp giúp trò chơi mà đề tài thực hiện có thể đem lại một phần trải nghiệm tốt dưới góc nhìn của báo cáo. Một lần nữa, xin cảm ơn cô Đào Thị Lệ Thuỷ và các nguồn kiến thức vì đã hiện thực hoá đề tài này!

**LỜI NÓI ĐẦU**

Nền khoa học công nghệ của thế giới ngày càng tiến bộ và phát triển, kéo theo đó là sự phát triển và thay đổi của nhiều các ngành nghề, lĩnh vực khác. Những sáng tạo của khoa học kĩ thuật được áp dụng vào nhiều mặt khác nhau của đời sống xã hội, giúp nâng cao năng suất, chất lượng, hiệu quả công việc và đem đến những cải tiến to lớn.

Ngành công nghiệp giải trí không nằm ngoài sự ảnh hưởng của khoa học công nghệ. Các loại hình giải trí hiện đại phát triển khi các thiết bị điện tử có giá thành phải chăng đến được với nhiều tệp khách hàng, trong số đó có ngành công nghiệp trò chơi điện tử. Những trò chơi điện tử qua thời gian có xu hướng đi lên về đồ hoạ, cơ chế, yêu cầu sức mạnh tính toán lớn hơn từ các thiết bị. Nhờ sự tiến bộ không ngừng của khoa học công nghệ mà các thiết bị điện tử đã có khả năng tính toán số lượng lớn với tốc độ nhanh, tạo điều kiện cho việc phát triển nhiều thể loại trò chơi như giải đố, thám hiểm, mô phỏng, thế giới mở,… và dạng trò chơi đi cảnh.

Với sức mạnh hiện tại của các thiết bị như điện thoại và máy tính, các trò chơi có thể thực hiện đủ các phép tính cần thiết với tốc độ cao để thực hiện nhiều chức năng như đồ hoạ đa màu sắc, tăng tốc độ khung hình, hoạt ảnh động, các phép tính mô phỏng vật lý,… để hiện thực hoá ý tưởng của các nhà phát triển và phát hành. Lựa chọn Java làm ngôn ngữ lập trình chính trong bài tập lớn môn học Công nghệ Java, em xin phép trình bày báo cáo “Lập trình trò chơi chủ đề đi cảnh bằng Java”. Với thiết kế ngôn ngữ lập trình hướng đối tượng cùng nhiều thư viện có sẵn giúp ích, việc tạo ra một trò chơi 2D với cơ chế chơi đơn giản là khả thi khi sử dụng Java.

**MỤC LỤC**

[**I. Khái niệm 4**](#_Toc164767164)

[**II. Thành phần cơ bản 4**](#_Toc164767165)

[**III. Vấn đề kĩ thuật 7**](#_Toc164767166)

[**IV. Cấu trúc chương trình 10**](#_Toc164767167)

[**V. Kết quả 14**](#_Toc164767168)

[**VI. Giải quyết vấn đề kĩ thuật 18**](#_Toc164767169)

[**VII. Đánh giá và cải thiện 23**](#_Toc164767170)

[**VIII. Tài liệu tham khảo 23**](#_Toc164767171)

# Khái niệm

Theo Wikipedia, trò chơi đi cảnh là một thể loại phụ của trò chơi hành động, có mục tiêu cốt lõi là di chuyển nhân vật đến các điểm trong môi trường màn chơi. Màn chơi có các địa hình không bằng phẳng với nhiều độ cao khác nhau, yêu cầu người chơi phải thao tác nhảy và leo trèo để đi tiếp.

Có nhiều cách triển khai mô hình trò chơi đi cảnh, báo cáo lựa chọn việc thiết lập lát cắt 2 chiều Oxy với thiết kế địa hình đặt trước, người chơi di chuyển trái phải trục x và nhảy hay rơi trên trục y áp dụng phép tính giả lập trọng lực. Các vật phẩm được xác đinh trước vị trí xuất hiện, các cơ chế được thiết lập sẵn trên nền bản đồ Oxy đã thiết kế từ trước. Trên phạm vi là một bài tập lớn, quy mô đề tài do báo cáo triển khai nằm ở mức nhỏ, không đầy đủ hoàn toàn mọi yếu tố của một trò chơi đi cảnh.

# Thành phần cơ bản

1. **Bản đồ màn chơi**

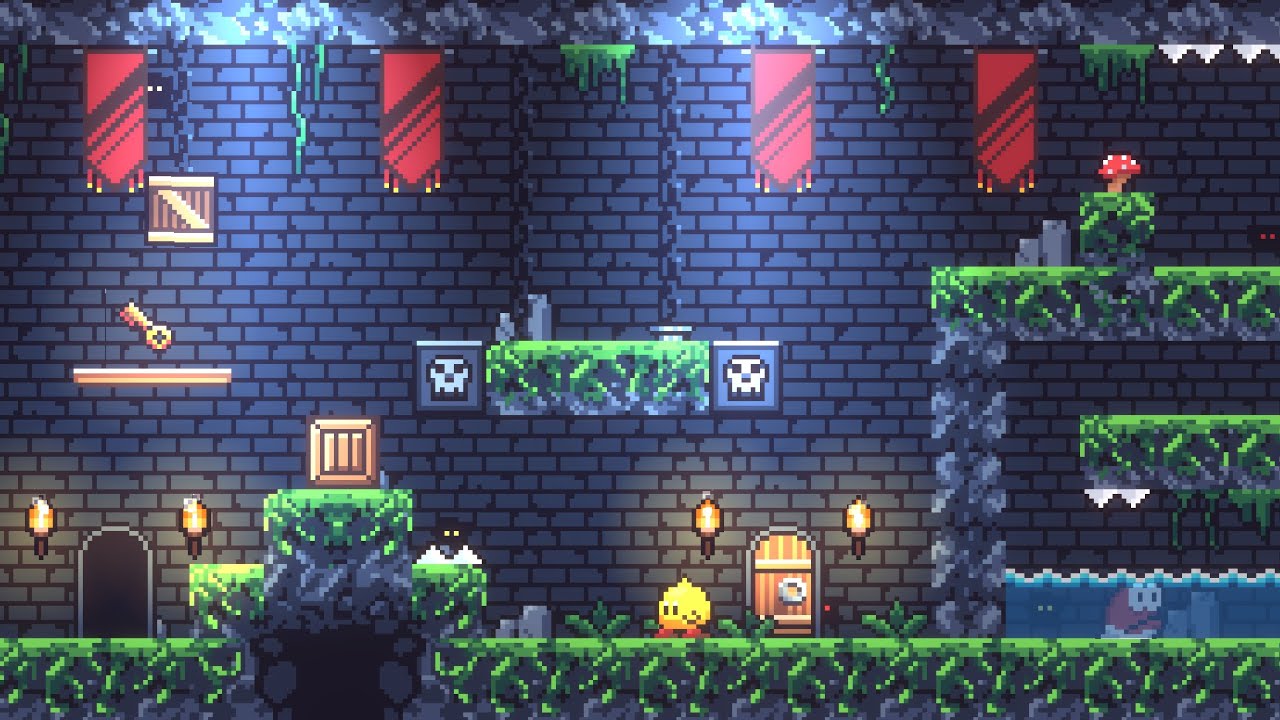
Màn chơi có bản đồ được cấu thành với cấu trúc tương tự mảng 2 chiều, nơi từng phần tử đại diện cho từng ‘ô vuông’ có kích thước xác định giống nhau thể hiện cho một loại vật liệu nhất định. Dựa vào thuộc tính của vật liệu, có thể quyết định khả năng giữa người chơi và ô vuông khi tiếp xúc với nhau.

Ví dụ về Bản đồ đồ hoạ và Bản đồ thuộc tính

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1** | **1** | **0** | **0** | **0** |
| **1** | **1** | **0** | **2** | **2** |
| **1** | **3** | **0** | **0** | **0** |
| **1** | **3** | **0** | **1** | **1** |
| **1** | **1** | **1** | **1** | **1** |



1. **Nhân vật của người chơi**

Người chơi có thể điều khiển nhân vật là một thực thể di chuyển theo lệnh điều khiển từ bàn phím và chịu ảnh hưởng của trọng lực. Người chơi có thể di chuyển trong các khối không và không thể đi xuyên qua các khối đặc. Một số khối đặc biệt như thang hay khối 1 có thể đi xuyên qua và có thao tác di chuyển trên chúng.

Nhân vật điều khiển được trong một bản đồ game 2D. (Nguồn: Blobby's Quest: 2D Platformer)

1. **Vật phẩm**

Vật phẩm xuất hiện tại vị trí chỉ định trong bản đồ màn chơi, sẽ được thu thập vào kho đồ khi người chơi chạm vào và được sử dụng để kích hoạt cơ chế hoặc thay đổi chỉ số của người chơi.

Nhân vật đứng giữa các vật phẩm trong game (Nguồn: Tynker)

1. **Khối cơ chế**

Trong bản đồ màn chơi có thể có các khối có thể tương tác. Các khối này có thể yêu cầu hoặc không yêu cầu người chơi có vật phẩm nhất định để kích hoạt khi tiếp xúc với người chơi.

Một khối cơ chế yêu cầu vật phẩm "Cypher" (Nguồn: Sea of Stars)

1. **Các yếu tố khác**

Trò chơi đi cảnh không chỉ dừng lại ở việc di chuyển người chơi. Có rất nhiều yếu tố có thể kết hợp cùng một trò chơi đi cảnh để tạo ra một trải nghiệm trò chơi đi cảnh có chiều sâu và làm người chơi hài lòng. Có thể kể đến như các hiệu ứng âm thanh, các NPC củng cố cốt truyện, kẻ địch có thể đánh bại, mở rộng hành động người chơi, các phần minigame làm tăng sự đa dạng về hình thức,…

Tương tác giữa người chơi và NPC. (Nguồn: YT - BrashMonkey)

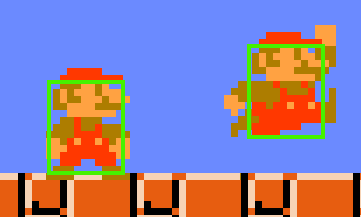
Kẻ địch khi thi triển chiêu thức. (Nguồn: Hollow Knight)



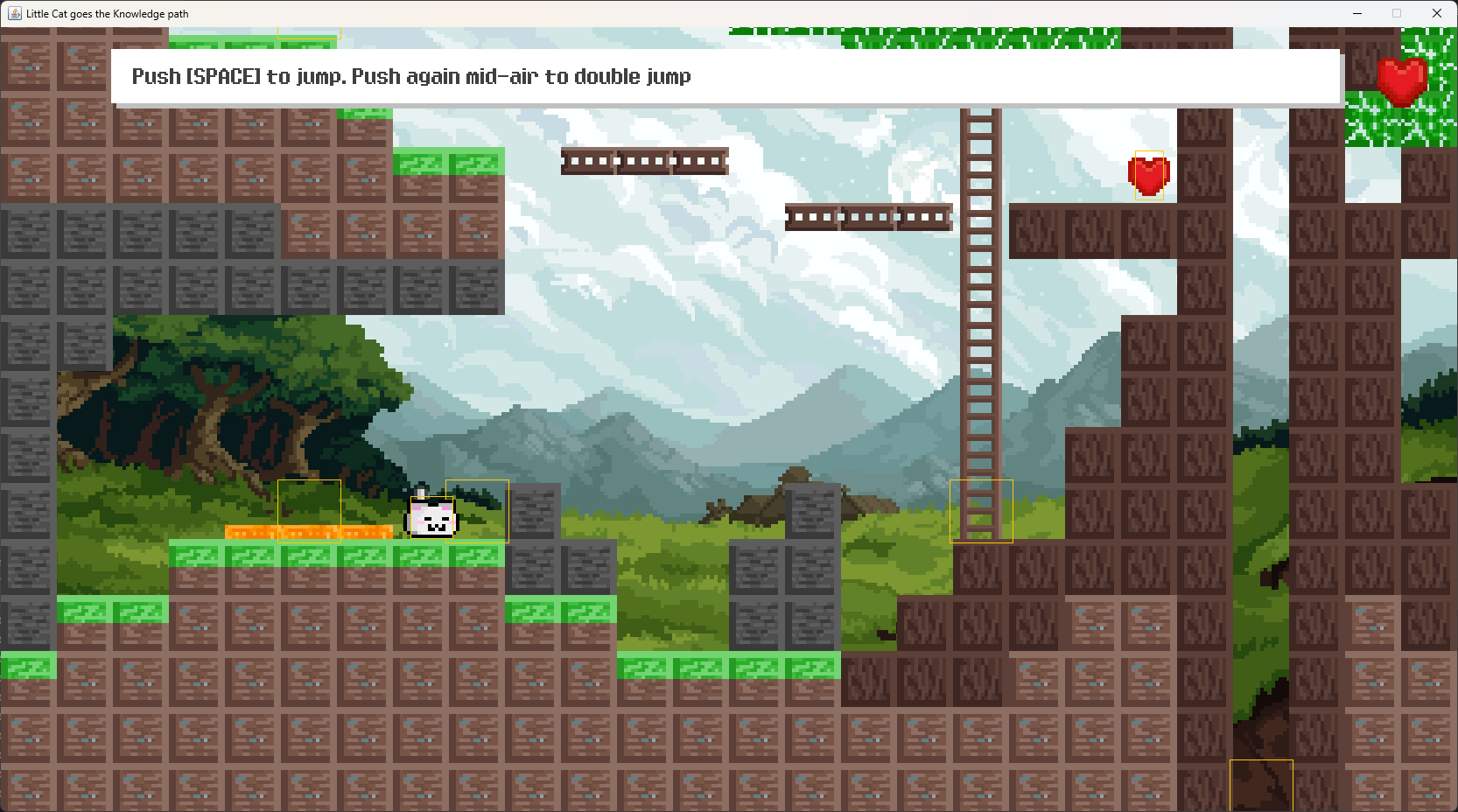
Minigame giải đố trong quá trình chơi (Nguồn: Rocco's Island)

Kỹ năng đặc biệt của người chơi. (Nguồn: Hollow Knight)

1. **Vấn đề kĩ thuật**
2. **Xử lý tiếp xúc của vật thể, môi trường**

Các vật thể, đặc biệt là người chơi, di chuyển và tương tác liên tục trong không gian cửa sổ trò chơi. Việc tiếp xúc giữa các vật thể với nhau và vật thể với môi trường là cần thiết để tính toán chính xác vị trí, hành động cần thiết khi người chơi thao tác trong trò chơi. Để xử lý vấn đề này, mỗi vật thể có thể tương tác tồn tại một khung chữ nhật đóng vai trò xác định trạng thái tiếp xúc, thường gọi là “hitbox”.

Mô tả về hitbox (khung chữ nhật màu xanh) của nhân vật trong trò chơi "Mario"

Hitbox thường có hình dạng cơ bản là một khung chữ nhật đi cùng vật thể. Trong Java hỗ trợ thư viện Rectangle về hình chữ nhật giúp tính toán về các vấn đề như kích thước, chứa, giao nhau,… trong hình học 2D. Sử dụng thư viện này, ta có thể dễ dàng đặt và tính toán trạng thái của các hitbox với nhau nhằm xác định thời điểm thực hiện cơ chế tương ứng.

Người chơi và khối cơ chế có hitbox giao nhau

Tuy nhiên, việc sử dụng hai hitbox để tính toán tiếp xúc chỉ phù hợp ở quy mô nhỏ với số lượng ít (như nhân vật, khối cơ chế, vật phẩm). Đối với tiếp xúc vật lý trong không gian màn chơi (mặt đất), sử dụng giao tiếp 2 hitbox sẽ khiến chương trình phải tính toán số lượng lớn, gây giảm hiệu suất.

Đối với tiếp xúc vật lý môi trường, ta lấy 4 góc của khung hitbox, quy đổi chúng sang toạ độ ô bản đồ và kiểm tra thuộc tính của ô để quyết định.

Các góc của hitbox người chơi khi trong khối đi qua được (đỏ) và không đi qua được (xanh dương)

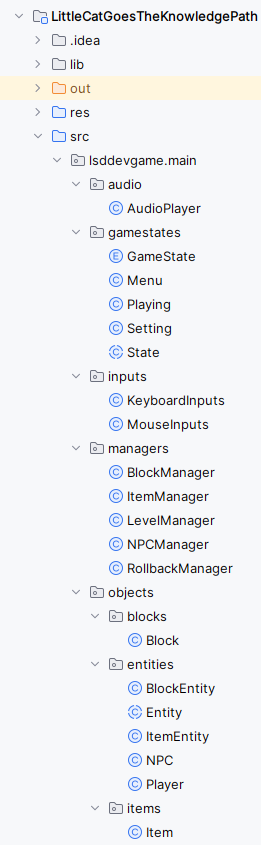
1. **Khung hình di chuyển cùng người chơi**

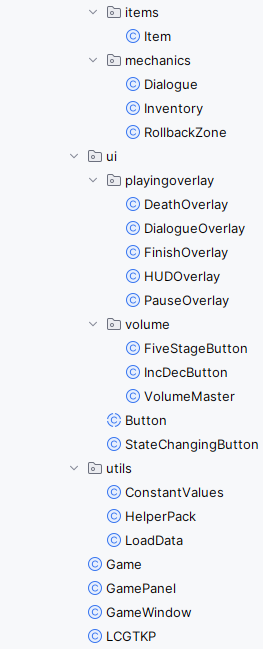
Đối với những màn chơi có bản đồ lớn hơn số ô định sẵn trên cửa sổ, việc hiển thị toàn bộ sẽ khiến người chơi không thể nhìn rõ chi tiết vì kích thước nhỏ, khiến người chơi khó chịu. Giữ ổn định kích thước hiển thị và chỉ hiển thị những vùng lân cận vị trí người chơi đang đứng sẽ giải quyết được vấn đề này.

Để thực hiện được chức năng này cần 2 biến số thay đổi toạ độ in trục x và y cho tất cả yếu tố đồ hoạ trong bản đồ màn chơi. Đặt giới hạn tại 4 khu vực cận viền của cửa sổ, khi người chơi tiếp cận các khu vực này, giá trị của hai biến số sẽ thay đổi và hình ảnh sẽ được xuất lệch theo sự điều chỉnh giá trị biến số. Toạ độ trong bản đồ là ổn định, chỉ có toạ độ kết xuất của đồ hoạ là thay đổi.

Thay đổi về giá trị toạ đô y khi xuất đồ hoạ sau khi vượt qua giới hạn

1. **Cấu trúc chương trình**

Trò chơi được lập trình theo kỹ thuật lập trình hướng đối tượng. Cấu trúc chương trình như ảnh sau:



* Folder **res**: Chứa tài nguyên hình ảnh, âm thanh và dữ liệu màn chơi của trò chơi
* Folder **src**: Chứa các package gồm các class ngôn ngữ Java là nền tảng của chương trình.
* Package **audio** : Chứa các class quản lý âm thanh trong trò chơi
  + Class **AudioPlayer**: Tải các file âm thanh cần thiết vào chương trình và thực hiện phát khi có lệnh
* Package **gamestates** : Chứa các phần quản lý trạng thái trò chơi
  + Enum **GameState**: Giữ biến xác định trạng thái trò chơi
  + Abstract class **State**: Khung của các trạng trái trò chơi
  + Class **Menu**: Quản lý màn hình chính (extends **State**)
  + Class **Playing**: Quản lý màn hình chơi (extends **State**)
  + Class **Setting**: Quản lý màn hình cài đặt (extends **State**)
* Package **inputs**: Chứa các class thực hiện tiếp nhận thông tin từ chuột và bàn phím
  + Class **KeyboardInputs**: Tiếp nhận thông tin nhập vào từ bàn phím
  + Class **MouseInputs**: Tiếp nhận thông tin nhập vào từ chuột
* Package **managers**: Chứa các class quản lý dữ liệu của màn chơi
  + Class **LevelManager**: Quản lý và xử lý tổng quát các vấn đề về dữ liệu, tương tác, cập nhật, đồ hoạ của màn chơi
  + Class **BlockManager**: Quản lý các khối của màn chơi
  + Class **ItemManager**: Quản lý các vật phẩm của màn chơi
  + Class **NPCManager**: Quản lý các NPC của màn chơi
  + Class **RollbackManager**: Quản lý các khu vực có thể khôi phục trạng trái gốc của màn chơi
* Package objects: Chứa các class lưu trữ đối tượng trong trò chơi
  + Package **blocks**: Chứa class lưu trữ dữ liệu loại khối
    - Class **Block**: Lưu trữ thông tin cơ bản về khối
  + Package **items**: Chứa class lưu trữ dữ liệu loại vật phẩm
    - Class **Item**: Lưu trữ thông tin cơ bản về vật phẩm
  + Package **entities**: Chứa các class lưu trữ thông tin về vật thể trong màn chơi
    - Abstract class **Entity**: Khung của các vật thể trong màn chơi
    - Class **Player**: Lưu trữ thông tin về người chơi, xử lý phương thức di chuyển của người chơi, tính toán vật lý lên người chơi, quản lý hoạt ảnh người chơi (extends **Entity**)
    - Class **BlockEntity**: Lưu trữ thông tin về các khối cơ chế trong màn chơi (extends **Entity** implements **Cloneable**)
    - Class **ItemEntity**: Lưu trữ thông tin về các vật phẩm có thể nhặt được trong màn chơi (extends **Entity**)
  + Package **mechanics**: Chứa các class lưu trữ thông tin của các đối tượng cơ chế đặc biệt trong màn chơi
    - Class **Inventory**: Lưu trữ và quản lý dữ liệu kho đồ của người chơi
    - Class **Dialogue**: Lưu trữ dữ liệu hội thoại của NPC trong màn chơi
    - Class **RollbackZone**: Lưu trữ thông tin và dữ liệu của khu vực có thể khôi phục trạng thái gốc trong màn chơi
* Package **ui**: Chứa các class quản lý và xử lý thao tác trên giao diện trò chơi
  + Abstract class **Button**: Khung của các nút bấm giao diện trò chơi
  + Class **StateChangingButton**: Lưu trữ và xử lý các nút bấm dạng thay đổi trạng thái trò chơi (extends **Button**)
  + Package **playingoverlay**: Chứa các class quản lý giao diện trong màn hình chơi
    - Class **DeathOverlay**: Quản lý và xử lý lớp phủ màn hình khi nhân vật không còn mạng
    - Class **FinishOverlay:** Quản lý và xử lý lớp phủ màn hình khi hoàn thành trò chơi
    - Class **PauseOverlay**: Quản lý và xử lý lớp phủ màn hình khi dừng màn chơi
    - Class **HUDOverlay**: Xử lý hiển thị lớp phủ thông tin mạng, kho đồ và khung thoại
    - Class **DialogueOverlay**: Xử lý hiển thị và quản lý thao tác khi hội thoại giữa người chơi và NPC
  + Package **volume**: Chứa các class quản lý phân mục điều chỉnh âm lượng trò chơi
    - Class **VolumeMaster**: Nhận thao tác của người chơi và điều chỉnh thông số âm lượng của **AudioPlayer**
    - Class **FiveStageButton**: Lưu trữ và xử lý nút điều chỉnh âm lượng 5 nấc trong **VolumeMaster** (extends **Button**)
    - Class **IncDecButton**: Lưu trữ và xử lý nut điều chỉnh âm lượng dạng thanh trong **VolumeMaster** (extends **Button**)
* Package **utils**: Chứa các giá trị mặc định của trò chơi và các class hỗ trợ quá trình xử lý trong trò chơi
  + Class **ConstantsValues**: Chứa các giá trị mặc định của trò chơi, người chơi, phương hướng, mặc định trạng thái.
  + Class **HelperPack**: Chứa các hàm xử lý sử dụng trong việc đảm bảo vật lý vật thể, giao tiếp giữa nhân vật và màn chơi, giữa người chơi và giao diện
  + Class **LoadData**: Chứa các hàm hỗ trợ tải tài nguyên cần thiết cho các class khác
* Class **GameWindows**: Tạo và hiển thị cửa sổ trò chơi khi chạy
* Class **GamePanel**: Xử lý hiển thị đồ hoạ trò chơi lên cửa sổ
* Class **Game**: Khởi tạo và quản lý tổng thể trò chơi, thực hiện cập nhật khung hình và tốc độ tính toán
* Class **LCGTKP**: Khởi động trò chơi

# Kết quả

Báo cáo sau thời gian tìm hiểu và thực hiện cho ra kết quả:

* Trò chơi chủ đề đi cảnh đồ hoạ 2D mặt cắt ngang với giao diện đơn giản.
* Nhân vật có hoạt ảnh, có các cơ chế di chuyển tiêu biểu của thể loại.
* Có 3 màn chơi với các vật phẩm, cơ chế giải đố, NPC đơn giản
* Có các màn hình Menu, Playing, Setting với các tuỳ chọn cơ bản.
* Có nhạc nền và hiệu ứng âm thanh đơn giản
* Lưu trữ dữ liệu màn chơi tách biệt với lập trình sử dụng JSON

1. **Giao diện**

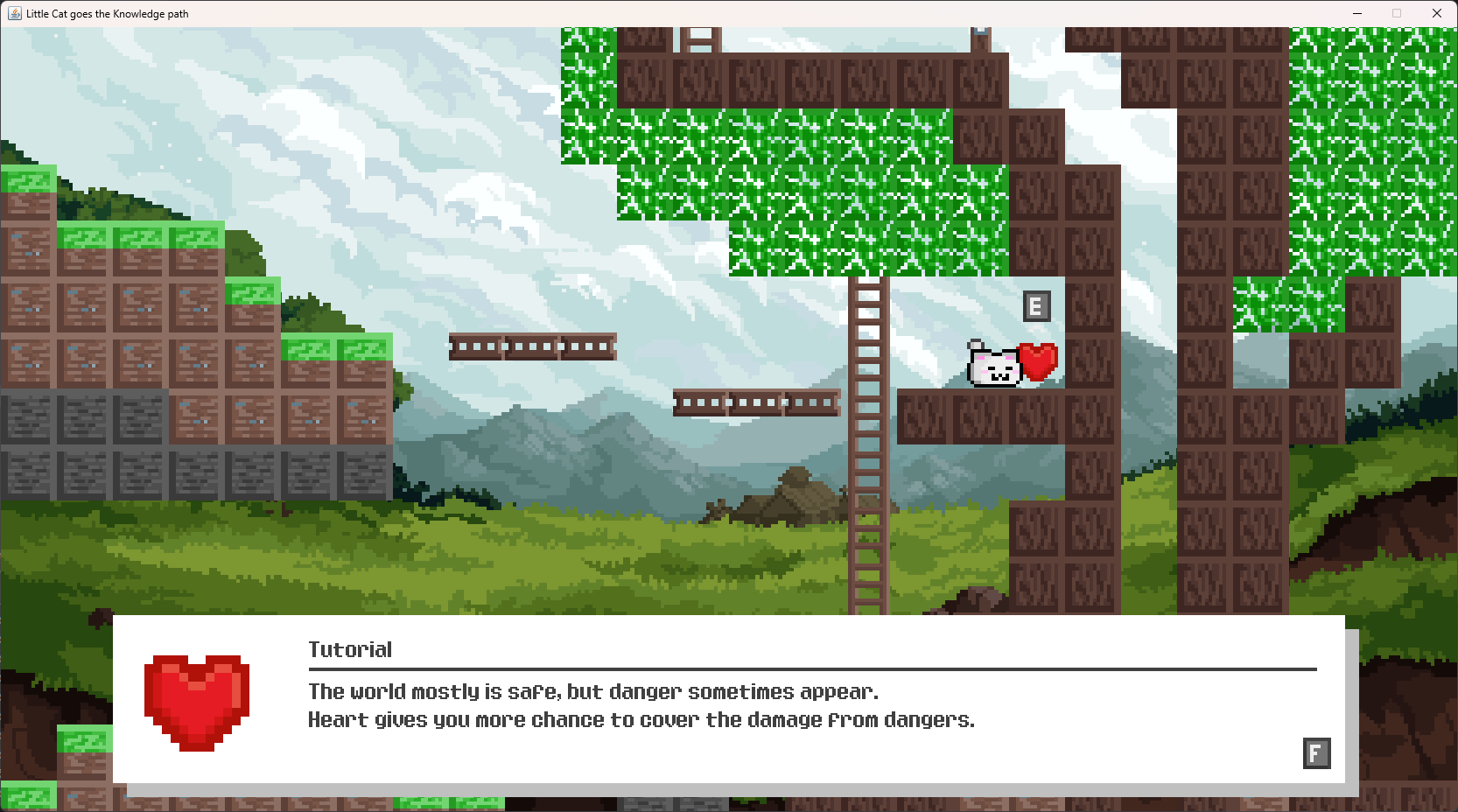
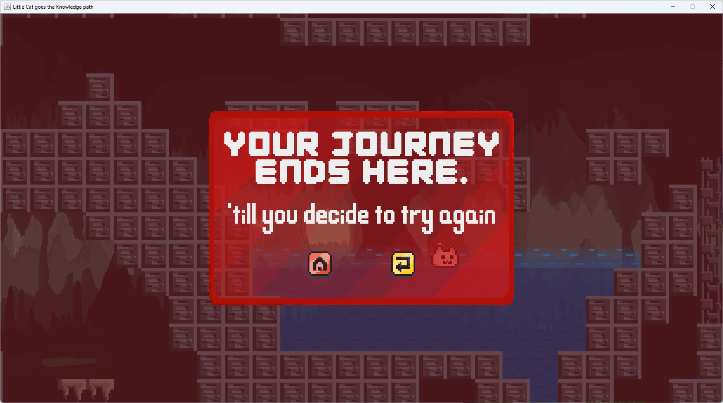
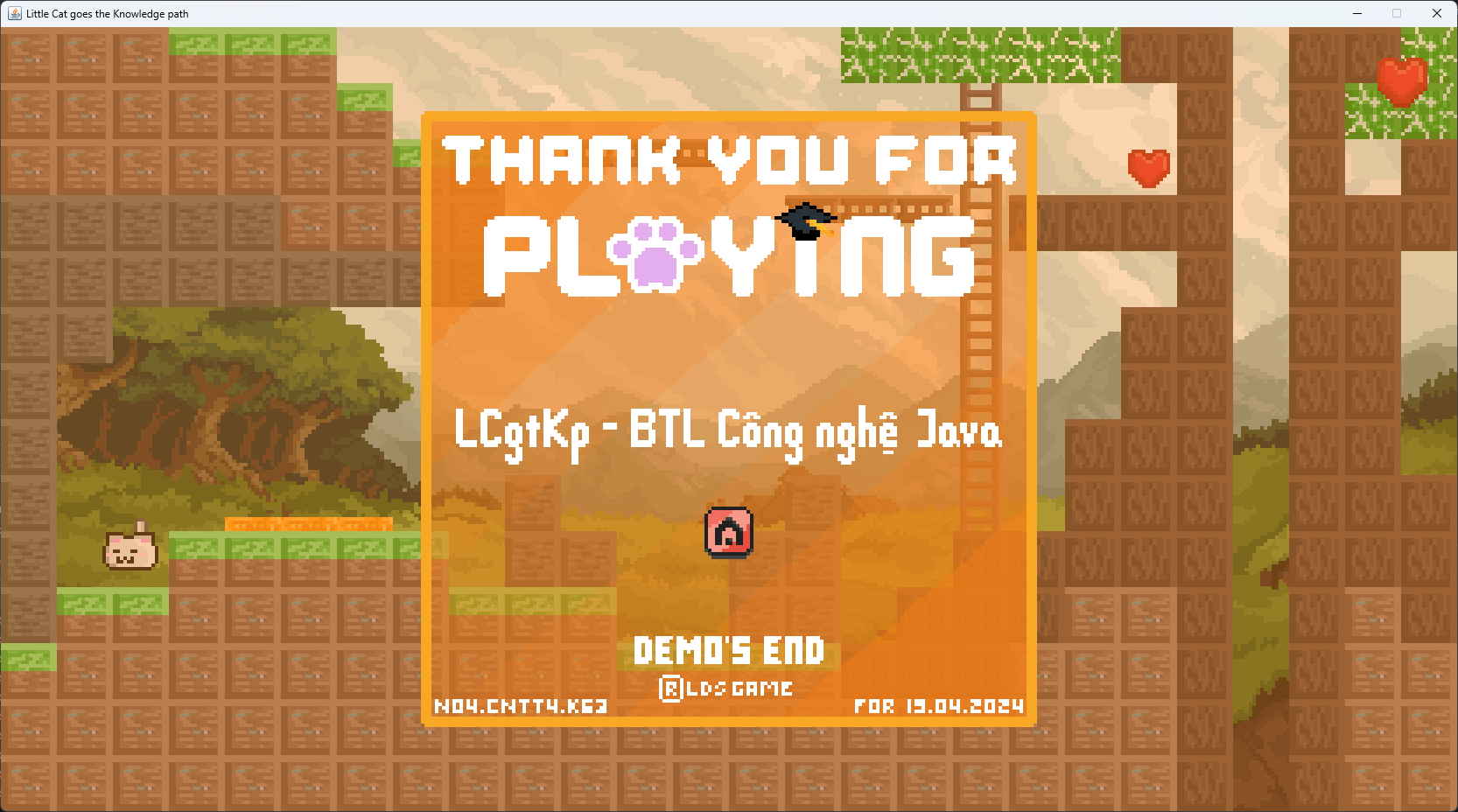
****

Màn hình chơi (Màn 1)

Lớp phủ khi trò chơi tạm dừng

Màn hình chính

Menu cài đặt

1. ******Di chuyển nhân vật**

Màn hình hội thoại giữa người chơi và NPC

Lớp phủ khi người chơi hoàn thành trò chơi

Lớp phủ khi người chơi "hết mạng"



Di chuyển sang phải (D)

Di chuyển sang trái (A)

1. **Tương tác vật phẩm và khối cơ chế**

3.1 Vật phẩm "Thêm mạng"

3.2 Vật phẩm "Chìa khoá"

Rơi từ khối 1 chiều (S + Space)

Nhảy (Space) và Nhảy kép (Space + Space)

Leo thang (W)

****

Người chơi nhấn E khi đang tiếp xúc gần, thực hiện đẩy hộp xuống

Khối cơ chế "Hộp đẩy được"

Người chơi tiếp xúc gần với mặt trái của khối

3.5 Tương tác khi có "chìa khoá"

3.3 Khối cơ chế "Cửa bị khoá"

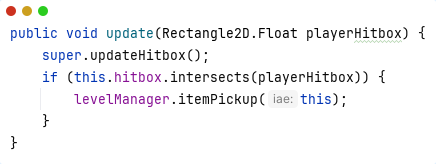
3.4 Thông báo khi tương tác không có "chìa khoá"

# Giải quyết vấn đề kĩ thuật

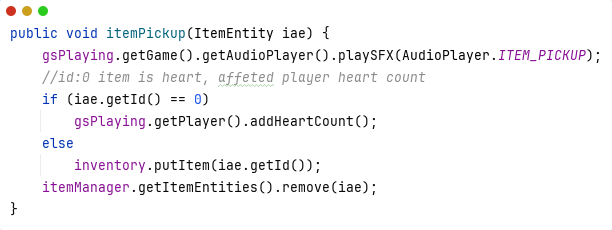
1. **Tiếp xúc vật thể, môi trường**

Để giải quyết vấn đề tiếp xúc giữa vật thể và vật thể, vật thể và môi trường, báo cáo ứng dụng kỹ thuật kiểm tra khung tiếp xúc của vật thể (hitbox). Mỗi vật thể dạng **Entity** được xác định một khung tiếp xúc dạng Rectangle2D.Float với các chỉ số x, y, w, h tuỳ chỉnh. Sử dụng chỉ số của khung này, có thể xác định các trạng thái tiếp xúc giữa vật thể - vật thể, vật thể - môi trường

Về vật thể - vật thể, lấy việc nhặt vật phẩm làm ví dụ:

* Với mỗi lần update() của levelManager, các vật phẩm trong màn chơi cũng được update cùng lúc:

Nhận tham số là hitbox của người chơi, khi hitbox của người chơi giao với hitbox của vật phẩm (hitbox.intersects()), vật phẩm sẽ gửi tín hiệu cho levelManager cho biết người chơi đã tiếp xúc với vật phẩm và cần phải thực hiện hành động cần thiết khi tiếp xúc với vật phẩm

* levelManager tiếp nhận tín hiệu từ vật phẩm và tiến hành thao tác cần thiết:

Việc người chơi tiếp xúc với vật phẩm được mặc định là hành động thu thập vật phẩm vào kho đồ của người chơi. levelManager trước tiên phát ra âm thanh nhặt vật phẩm thông qua audioPlayer. Kiểm tra ID của vật phẩm, levelManager sẽ quyết định hành động nhặt vật phẩm tương ứng với thu vật phẩm vào túi (iae.getId() != 0) để thêm vào kho đồ (inventory.putItem()) hay thêm mạng cho người chơi (iae.getId() == 0, trường hợp đặc biệt) để gửi tín hiệu thêm mạng đến nhân vật (gsPlaying.getPlayer().addHeartCount()). Sau khi hoàn thành hành động cần thiết, levelManager gửi tín hiệu xoá thực thể vật phẩm vừa nhặt trong bản đồ đến itemManager (itemManager.getItemEntities().remove()) tránh tình trạng một vật phẩm được thu thập liên tục.

Về vật thể - môi trường, lấy nhân vật, cụ thể là xử lý di chuyển ngang trục x làm ví dụ:

* Với mỗi lần update của gsPlaying trong trạng thái PLAY, nhân vật cũng được cập nhật cùng lúc, trong đó có cập nhật về toạ độ, bao gồm toạ độ x:
  + Trong player: player.update() => updatePosition() => updateXPos()

Sử dụng hàm hỗ trợ canCollisionWithCheck() từ HelperPack, player gửi các thông số của hitbox vào hàm để hàm tiến hành kiển tra tiếp xúc giữa nhân vật và môi trường trước khi đưa ra thay đổi về toạ độ x của nhân vật:

canCollisionWithCheck() gọi hàm kiểm tra isSolidCheck() cho từng góc của player hitbox (gồm trái trên, phải trên, trái dưới, phải dưới). isSolidCheck() quy đổi toạ độ góc thành toạ độ bản đồ và kiểm tra trại thái đặc (khi toạ độ vượt ngoài phạm vi cửa sổ, khi thuộc tính khối là đặc (LayerLevel.MIDDLE (= 1))). Nếu rơi vào trạng trái đặc, canCollisionWithCheck() trả về giá trị false tức không thể giao, ngược lại nếu có thể.

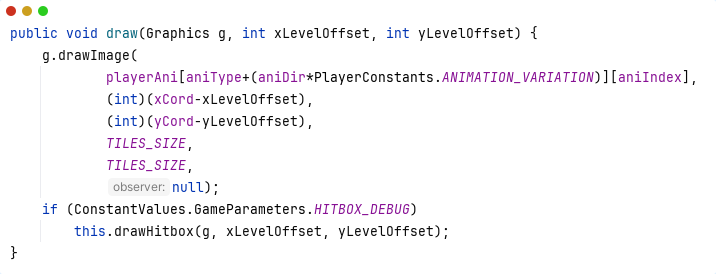
Nhân vật sau khi nhận được kết quả kiểm tra của canCollisionWithCheck() tiến hành quyết định hành động. Nếu có thể giao, toạ độ x sẽ được cập nhật thay đổi theo xSpeed (âm nếu đi qua trái, dương nếu đi qua phải). Trường hợp không thể, toạ độ x sẽ được đặt tại vị trí gần nhất với khối đặc mà không gây ra giao giữa cả 2 sử dụng giá trị từ GetWallNextToEntity() của HelperPack.

1. **Di chuyển khung hình theo người chơi**

Tất cả màn chơi có trong đề tài đều có kích thước lớn hơn cửa sổ màn chơi (mặc định 26\*14 ô kích thước 16x16 pixel chưa áp dụng SCALING, 1664x896 sau SCALING). Việc di chuyển khung hình theo người chơi là vấn đề bắt buộc cần phải giải quyết.

Việc tính toán độ lệch kết xuất khung hình được thực hiện trong Playing. Vì trọng tâm vấn đề kết xuất khung hình lệch là từ di chuyển nhân vật, độ lệch kết xuất được tính dựa hoàn toàn vào toạ độ người chơi:

Với mỗi lần cập nhật Playing, độ lệch cũng được cập nhật cùng lúc. Mỗi lần cập nhật, gsPlaying lấy giá trị toạ độ x và y của hitbox nhân vật. Độ lệch giữa toạ độ x của người chơi với độ lệch kết xuất x hiện tại được tính và lưu trong different. Nếu độ lệch này lớn hơn viền lệch phải hoặc nhỏ hơn viền lệch trái, tiến hành điều chỉnh độ lệch x sao cho toạ độ x của nhân vật + độ lệch phải nằm trong ô cho phép đã quy định. Đồng thời độ lệch được giữ trong khoảng từ 0 đến giá trị lệch tối đa cho phép (độ rộng màn chơi trừ viền lệch phải) để giữ kết xuất không xuất hình ảnh nằm ngoài bản đồ màn chơi. Tương tự như vậy, độ lệch y được cập nhật trong trường hợp vượt qua viền lệch trên hoặc viền lệch dưới sao cho tổng toạ độ y nhân vật và độ lệch y nằm trong ô cho phép. Giá trị của y được giữ trong phạm vi từ 0 đến giá trị lệch tối đa cho phép (độ cao màn chơi trừ viền lệch dưới).

Khi kết xuất hình ảnh của bất kì yếu tố nào trong màn chơi, ngoài tham số Graphics phục vụ xuất đồ hoạ, còn cần truyền vào 2 tham số độ lệch xLevelOffset và yLevelOffset, ví dụ từ nhân vật:

Trong khi toạ độ của nhân vật trong bản đồ (xCord, yCord) không thay đổi, quá trình kết xuất hình ảnh được điều chỉnh lệch theo giá trị xLevelOffset và yLevelOffset nhằm giữ nhân vật đảm bảo hiển thị trong phạm vi nhìn thấy của cửa sổ (0-1664 x, 0-896 y). Áp dụng độ lệch với tất cả yếu tố hình ảnh của bản đồ trong màn chơi, ta đạt được hiệu ứng camera di chuyển theo người chơi và giữ khung hình hiển thị trong khu vực người chơi đang thám hiểm.

# Đánh giá và cải thiện

Sản phẩm do báo cáo trình bày còn một số khuyết điểm:

1. Về nội dung

Chưa có một câu chuyện hoàn chỉnh, số lượng màn chơi ít, độ khó của cơ chế không nhiều, ít cơ chế đặc biệt.

1. Về kĩ thuật

Một số phần được lập trình chưa chuyên nghiệp, chưa hoàn toàn tuân thủ lập trình hướng đối tượng, nhiều phần chưa được tối ưu hoàn chỉnh, có thể gây ra tình trạng lag, bug và khó khăn trong việc mở rộng, nâng cấp về sau.

Trong tương lai, báo cáo có thể cải thiện:

* Sáng tạo thêm các bản đồ mới, cơ chế giải đố mới, tăng tính đa dạng
* Tiến hành sàng lọc và tối ưu dòng lệnh
* Thêm các yếu tố đặc trưng khác của trò chơi đi cảnh như kẻ địch, di chuyển lướt,…
* Cải thiện trải nghiệm người chơi bằng âm thanh, tăng hoạt hoạ, hình ảnh…

1. **Tài liệu tham khảo**

[1] Slide bài giảng của giảng viên **Đào Thị Lệ Thuỷ** – ĐH GTVT.

[2] Danh sách phát “*Platformer Tutorial - Java*”, tác giả **Kaarin Gaming** : https://youtube.com/playlist?list=PL4rzdwizLaxYmltJQRjq18a9gsSyEQQ-0&si=HVZjtBHdhav6QSIY