Shape, square

Description automatically generated**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI**

**UNIVERSITY OF TRANSPORST AND COMMUNICATIONS**

Logo

Description automatically generated

**BÁO CÁO BÀI TẬP LỚN BỘ**

**MÔN CÔNG NGHỆ JAVA**

***ĐỀ TÀI: LẬP TRÌNH GAME 2D***

**Giảng viên giảng dạy : (thầy) Vũ Huấn**

**Lớp : Công nghệ thông tin 6K62**

**Sinh viên thực hiện : Nguyễn Tiến Tùng**

**Mã sinh viên : 211200893**

**Hà Nội 2023**

**LỜI MỞ ĐẦU**

Hiện nay nhu cầu giải trí ngày càng phát triển, ngành công nghệ thông tin ngày càng đa dạng các lĩnh vực hơn. Trong đó ngành phát triển game là một trong những ngành càng ngày cảng phát triển, ngày càng nhiều game mới được ra mắt đáp ứng nhu cầu giải trí của con người. Để hiểu về nguyên lý, các bước phát triển một game em đã quyết định tự mình học và code ra một game 2D bằng ngôn ngữ lập trình java.

Chắc chắn rằng, với kinh nghiệm cũng như kiến thức còn non trẻ, nên game của mình vẫn còn nhiều thiếu sót và chưa hoàn thiện. Vì vậy em rất mong nhận được nhận xét từ thầy. Để game được thêm hoàn thiện.

Em xin trân thành cảm ơn thầy trong suốt quá trình giảng dạy đã truyền đạt cho em kiến thức để em có thể có thể thực hiện dự án của mình.

**Mục lục**

[**1. Giới thiệu tổng quan của trò chơi** 1](#_Toc134003878)

[**2. Giao diện game, ý tưởng code giao diện game** 1](#_Toc134003879)

[**2.1. Giới thiệu giao diện game** 1](#_Toc134003880)

[**2.2. Cách thức hoạt động của giao diện** 6](#_Toc134003881)

[**2.3. Giải thích cách vẽ giao diện** 9](#_Toc134003882)

[**2.3.1. Giải thích cách vẽ nhân vật và kẻ thù** 9](#_Toc134003883)

[**2.3.2. Giải thích các vẽ map** 11](#_Toc134003884)

[**2.3.3 Giải thích cách vẽ các trạng thái menu như pause game, menu chính, game over** 12](#_Toc134003885)

[**3. Mô hình các gói, các lớp của game** 14](#_Toc134003886)

[**4. Giải thích các logic game** 18](#_Toc134003887)

[**4.1 Logic di truyển của nhân vật và kẻ thù** 18](#_Toc134003888)

[**4.2. Logic gây sát thương của nhân vật và kẻ thù** 23](#_Toc134003889)

[**4.3. Logic chuyển trạng thái của game** 25](#_Toc134003890)

[**5. Kết Luận** 26](#_Toc134003891)

[**6. Tài liệu tham khảo** 26](#_Toc134003892)

# **1. Giới thiệu tổng quan của trò chơi**

Hiện tại trò chơi vẫn chưa có tên, nên em sẽ đặt tên cho game là Game Big hat knight. Sở dĩ game có tên như vậy xuất phát từ đặc điểm ngoại hình nhân vật, nhân vật xuất hiện với chiếc mũ to, mang trên mình thanh kiếm nên em đặt tên game là Big hat knight.

\* Game đáp ứng các yêu cầu:

- Thiết kế theo hướng đối tượng

- Làm việc với tệp tin ( đọc file ảnh, đọc file âm thanh)

- Thao tác với cơ sở dữ liệu, dùng My SQL để lưu vị trí nhân vật, máu nhân vật, map

- Đa luồng: có luồng âm thanh, luồng logic tạo vòng lặp game.

- Có bắt lỗi

- Có GUI

\* Các tính năng của trò chơi:

- Game bao gồm giao diện đồ hoạ thân thiện người dùng.

- Game có các đối tượng như: nhân vật, kẻ thù, map, các item tăng máu, các menu trạng thái game.

- Game có kết thúc game: ở mỗi màn chơi thì sẽ có một map bản đồ riêng, có số lượng kẻ thù khác nhau, sau khi nhân vật tiêu diệt xong kẻ thù thì sẽ kết thúc màn và sẽ được chuyển sang màn mới. Game được thiết kế có 3 màn. Sau khi kết thúc màn cuối thì game sẽ kết thúc.

- Game có âm thanh, các màn khác nhau có âm thanh khác nhau, từng trạng thái của nhân vật đều có một loại âm thanh khác nhau.

- Game có chức năng lưu game, khi chúng ta kich vào nút lưu game thì dữ liệu về toạ độ nhân vật, máu, vị trí nhân vật trong map lên SQL. Lần sau khi game chạy lên ta có thể tiếp tục chơi.

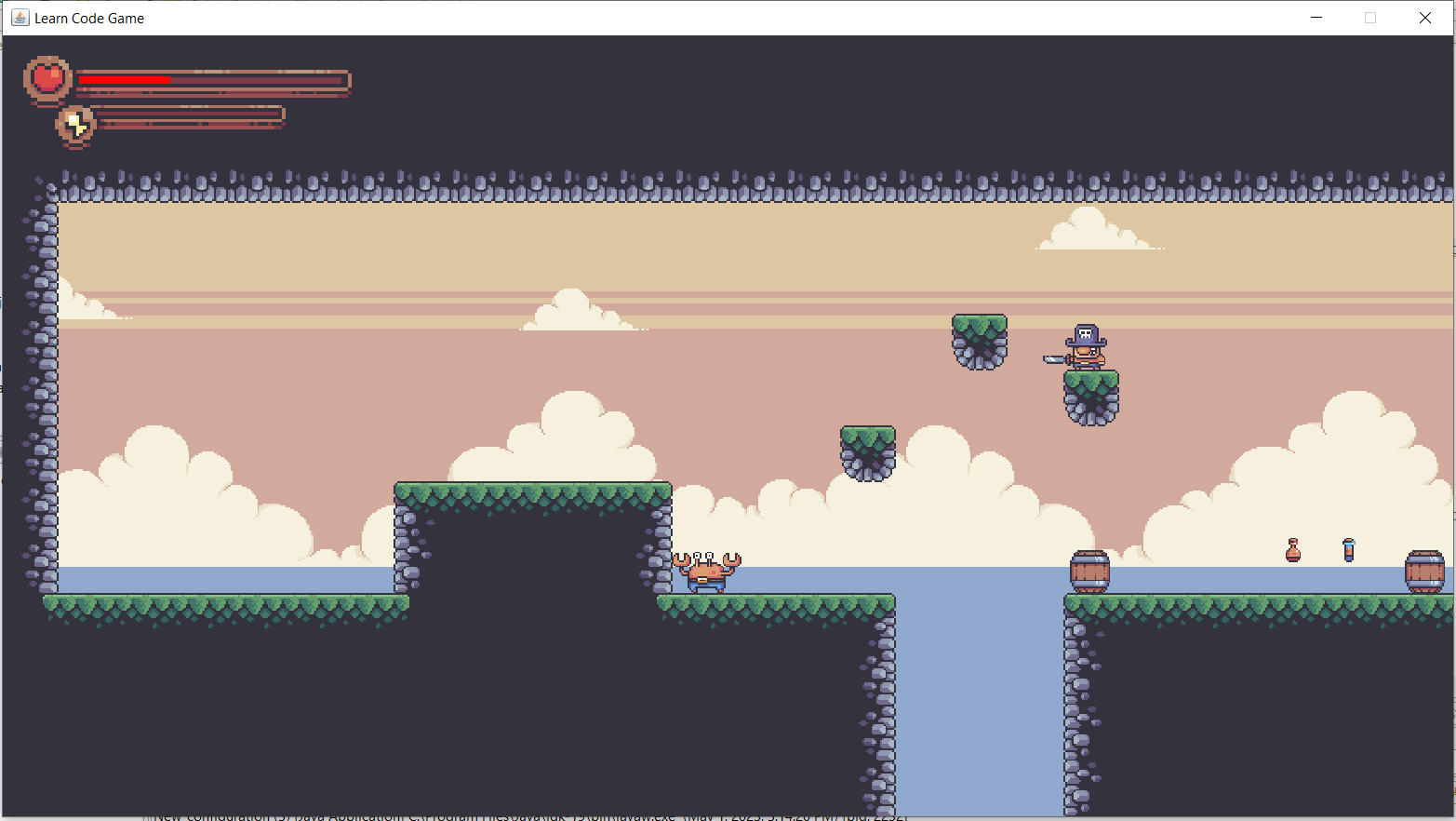
# **2. Giao diện game, ý tưởng code giao diện game**

## **2.1. Giới thiệu giao diện game**

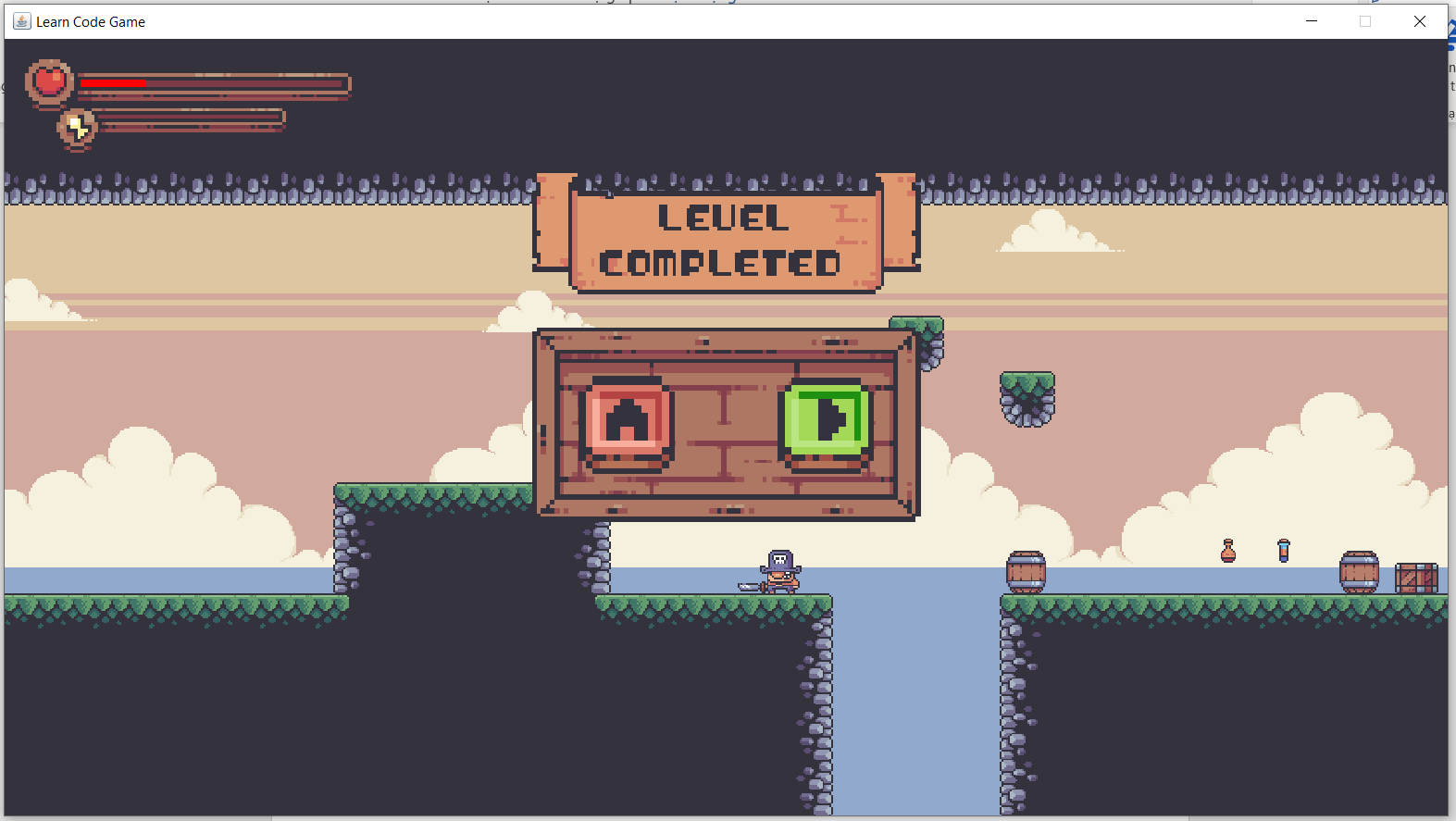
Giao diện trò chơi gồm có các màn, nhân vật, kẻ thù, và các menu trạng thái của game.



Giao diện Menu chính



Giao diện map 1

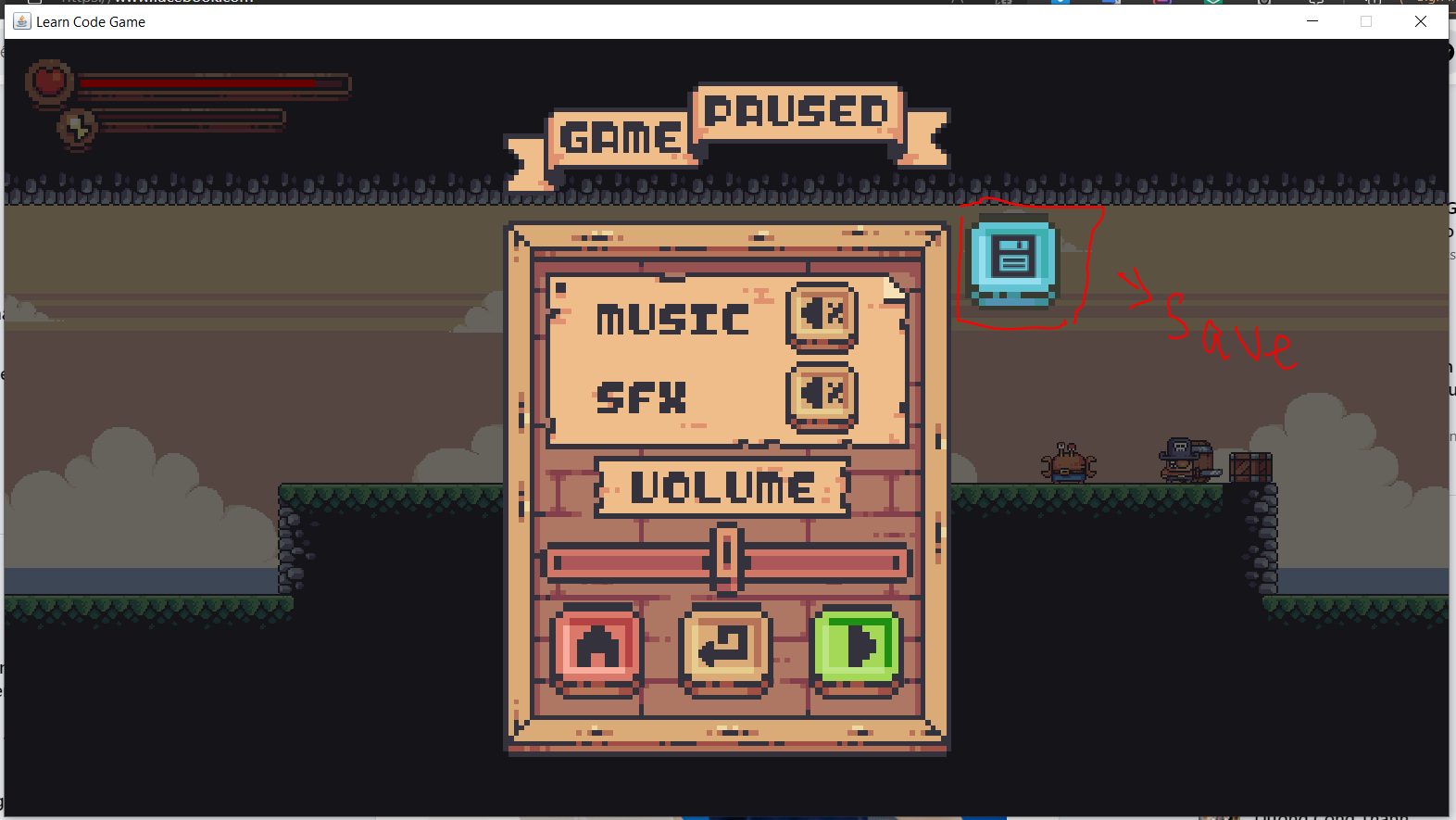


Menu hoàn thành map 1

Ảnh có chứa văn bản, ảnh chụp màn hình

Mô tả được tạo tự động

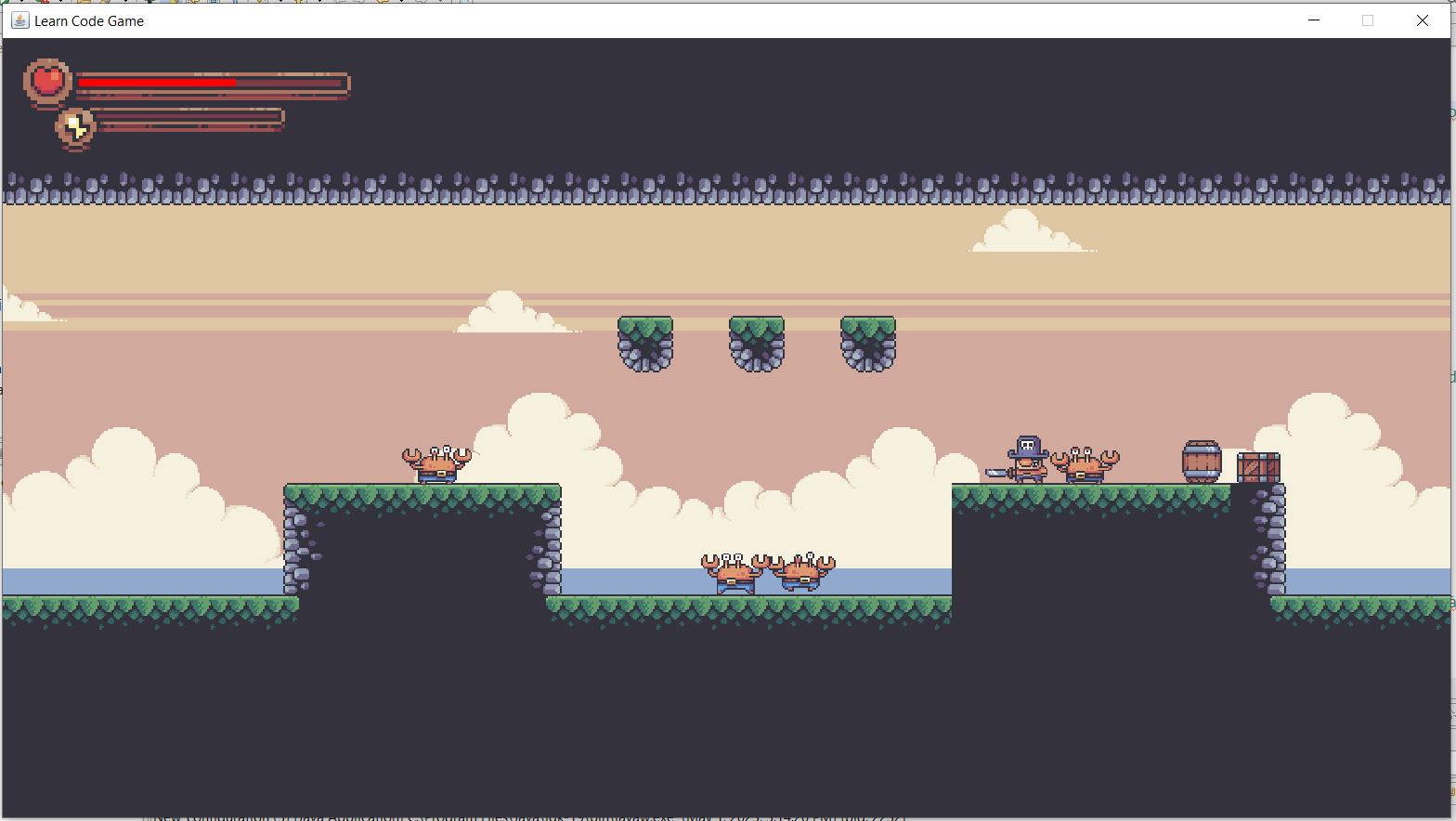
Giao diện map 2



Giao diện Pause Game



Menu khi nhân vật chết



Giao diện map 3



Giao diện game Option

## **2.2. Cách thức hoạt động của giao diện**



Giao diện Menu chính

*a) Giao diện Menu chính*

Khi bắt đầu vào game thì Menu chính của game hiện ra, tại đây có các lựa chọn Continue, Newgame, Option, Quit.

Nếu chúng ta nhấn continue thì chúng ta sẽ tiếp tục chơi game tại vị trí mà ta đã lưu từ trước.

Nếu chúng ta nhấn NewGame thì toàn bộ game của chúng ta sẽ reset lại từ đầu. Từ máu nhân vật, vị trí nhân vật, map.

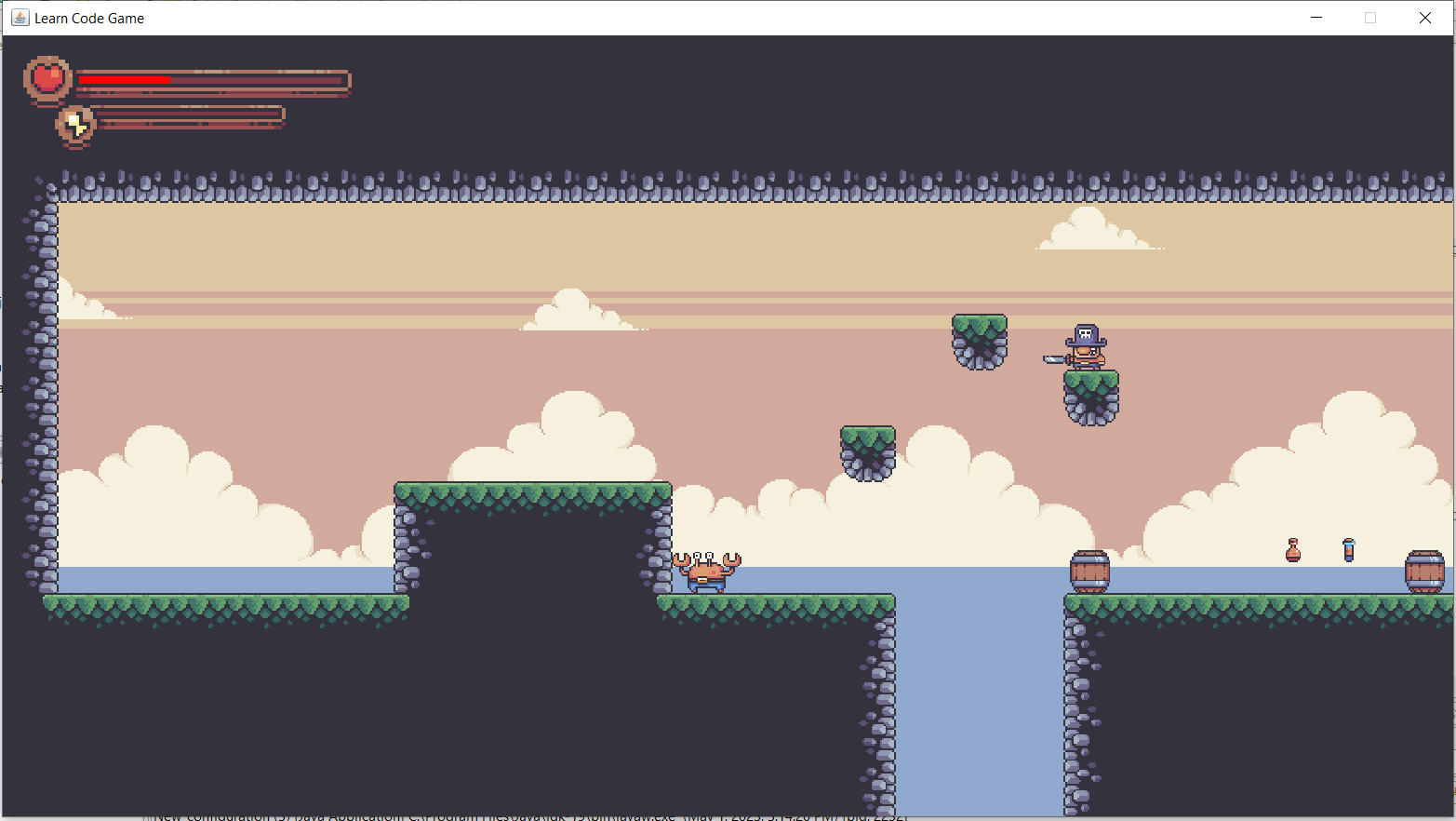
Nếu chúng ta nhấn vào Option thì bảng cài đặt sẽ hiện ra tại đây chúng ta có thể tuỳ chỉnh âm thanh có thể bật tắt, điều chỉnh âm lượng, bặt tắt âm thanh hiệu ứng nhân vật bằng các tương tác với button SFX, bật tắt âm thanh nền bằng cách tương tác với button music. Thay đổi âm lượng bằng cách kéo thả volume.



Giao diện game Option

Nếu chúng ta nhấn vào Quit thì sẽ thoát game.

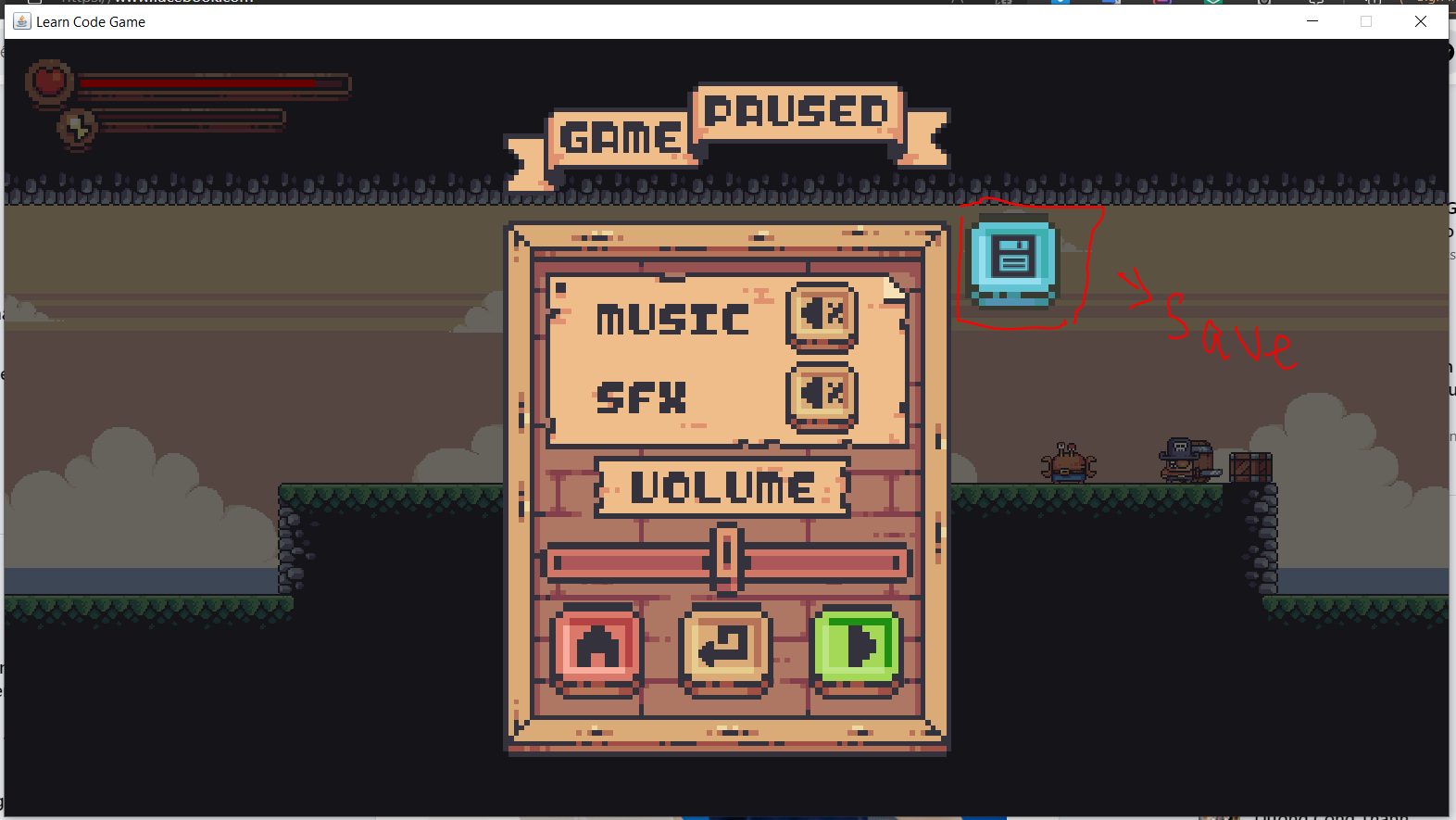
b) Giao diện các màn, pause game và hoàn thành màn chơi, game over



Giao diện map 1

Mỗi một màn thì sẽ có map bản đồ cho các màn, trong màn sẽ có kẻ thù vào các item tăng máu cho nhân vật.

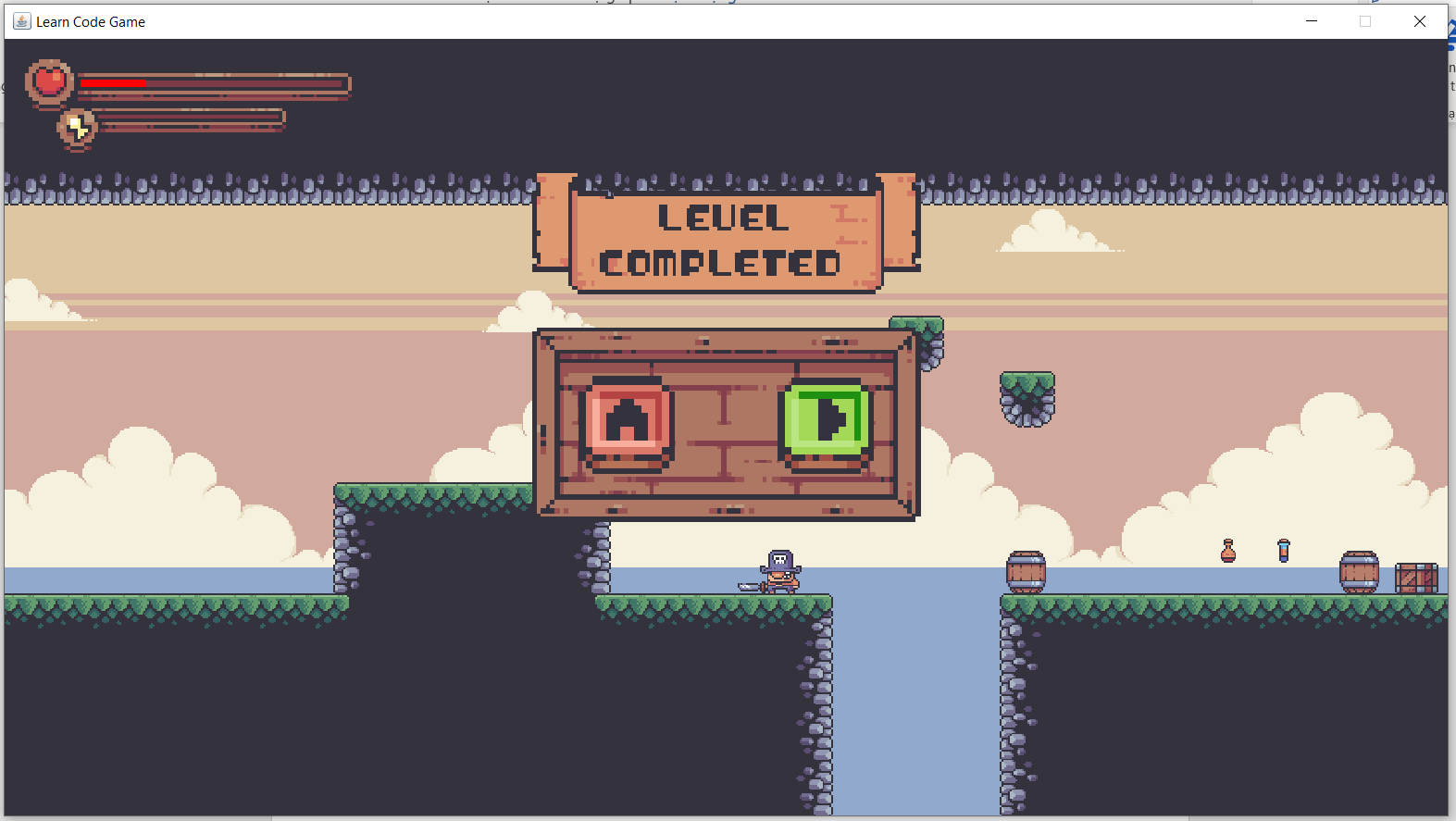
Trong mỗi màn nếu chúng t muốn tạm ngừng thì có thể nhấn phím Escapse và giao diện PauseGame hiển thị



Giao diện Pause Game

Tại đây người chơi có thể thay đổi trạng thái âm thanh, có nút tiếo tục chơi, có nút chơi lại từ đầu của màn này, có nút quay trở lại Menu chính, có nút save game.

Sau khi nhân vật chém chết hết kẻ thù trong màn thì hiển thị một bảng Menu với hai lựa chọn đi tới màn tiếp hoặc quay trở về Menu chính.



Menu hoàn thành map 1

Nếu như nhân vật để kẻ thù giết chết thì sẽ hiển thị giao diện game over

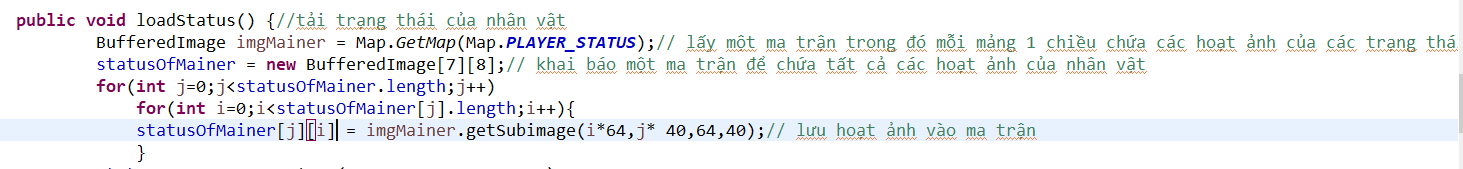


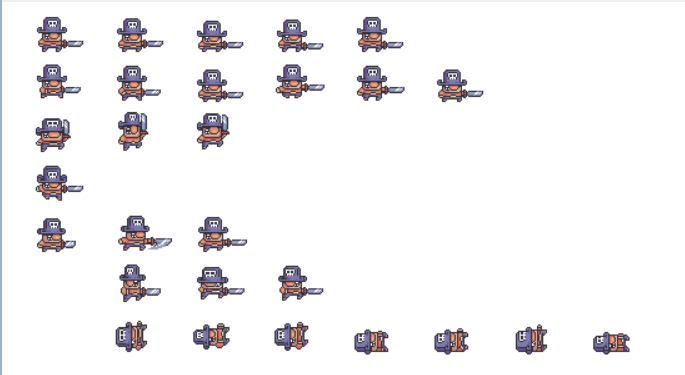
Menu khi nhân vật chết

## **2.3. Giải thích cách vẽ giao diện**

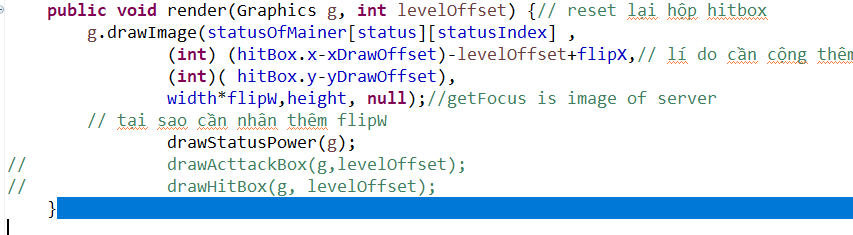
### **2.3.1. Giải thích cách vẽ nhân vật và kẻ thù**

B1: Ta lưu các hoạt ảnh trạng thái của nhân vật, kẻ thù vào mảng ảnh 2 chiều. Trong đó mảng hai chiều này một chiều có chỉ số là trạng thái của nhân vật một chiều là dòng trạng thái của nhân vật





B2: Ta cập nhật các trạng thái của nhân vật tuỳ thuộc vào các tương tác sự kiện bàn phím, chuột của người dùng



Tại một tương tác sự kiện thì sẽ truyền vào các key Input tại mỗi key input ta sẽ thiết lập các biến logic trạng thái tương ứng cho nhân vật.

Sau đó chỉ số StatusIndex sẽ được chạy và hoạt ảnh của nhân vật được vẽ ra. Như vậy hoạt ảnh của nhân vật sẽ được vẽ liên tục cho từng trạng thái. Nhân vật luôn ở trạng thái động.

Ảnh có chứa văn bản

Mô tả được tạo tự động

Chạy các hoạt ảnh trong 1 dòng trạng thái của nhân vật

Ví dụ: khi ta nhấn nhảy thì biến trạng thái mà chúng ta thiết lập cho trạng thái của nhân vật là nhảy sẽ chuyển thành true, sau đó trạng thái của nhân vật sẽ sẽ truyển thành trạng thái nhảy và status sẽ được chuyển sang trạng thái nhảy tức giá trị của status = 2

Ảnh có chứa văn bản

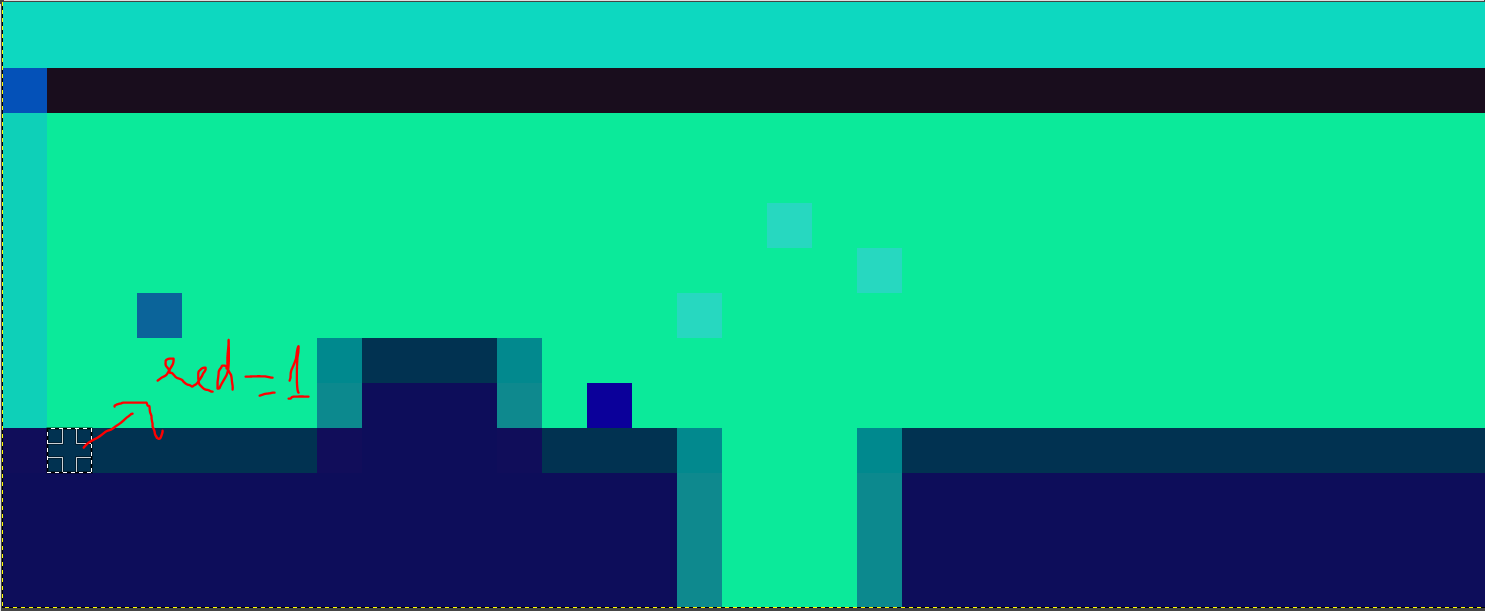
Mô tả được tạo tự động

### **2.3.2. Giải thích các vẽ map**

Em dùng phương pháp vẽ map dựa vào mảng chỉ số màu của các ô. Màu xanh em định danh cho kẻ thù, màu đỏ em vẽ map. Nguyên lý của phương pháp này đó là em định danh cho tất cả các phần từ của map là một số nguyên. Sau đó em sẽ vẽ một lưới ma trận các ô pixel và tuỳ thuộc vào vị trí nào trong ma trận chúng ta sẽ đặt các phần tử map như thế nào mà em sẽ thiết lập màu cho các ô pixel như vậy. Ví dụ các phần tử trên nền của game em sẽ để giá trị màu đỏ của nó là 1. Thì em sẽ vẽ các ô trên nền có giá trị màu đỏ là 1.

Ảnh có chứa văn bản, vải vóc

Mô tả được tạo tự động



Ảnh có chứa văn bản

Mô tả được tạo tự động

Ảnh có chứa văn bản

Mô tả được tạo tự động

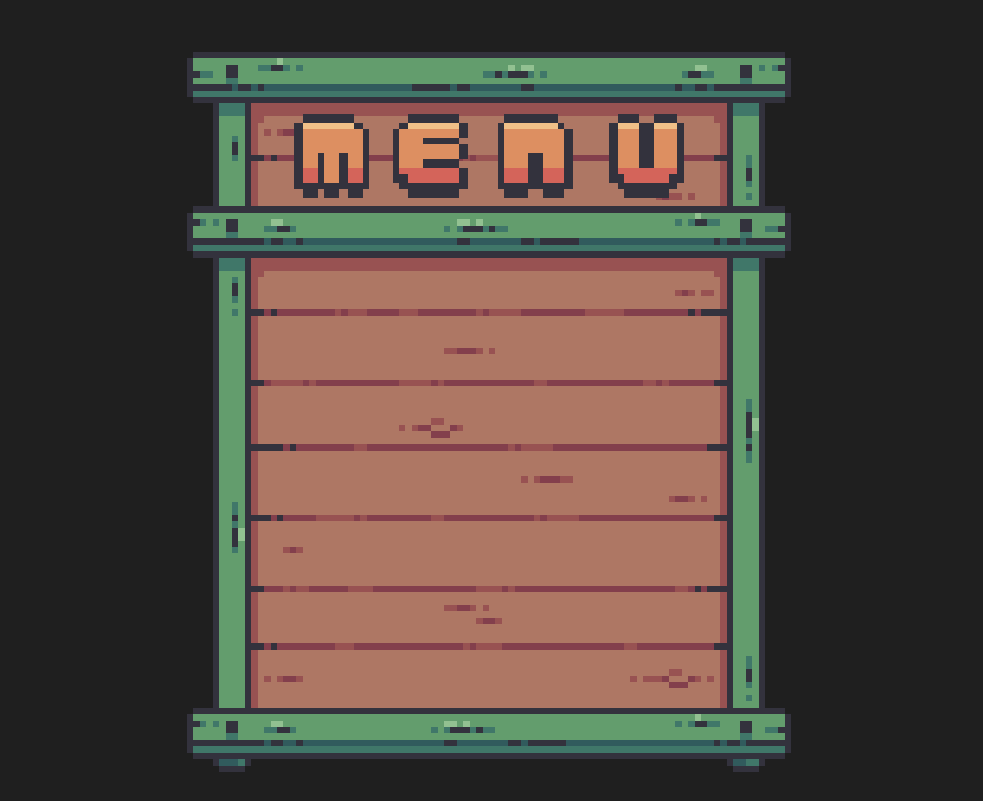
Ảnh có chứa văn bản

Mô tả được tạo tự động

Thiết lập kẻ thù cho Map

### **2.3.3 Giải thích cách vẽ các trạng thái menu như pause game, menu chính, game over**

Ta dùng các bức ảnh có sẵn rồi dùng phương thức draw trong thư viện GUI để vẽ các bức ảnh.

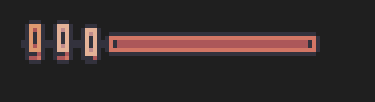
Dưới đây là một số bức ảnh cần cho quá trình thiết kế các menu trạng thái của game.Ảnh có chứa văn bản, biển hiệu

Mô tả được tạo tự động

Ảnh có chứa văn bản, biển hiệu, bảng điểm

Mô tả được tạo tự độngẢnh có chứa biểu đồ

Mô tả được tạo tự động

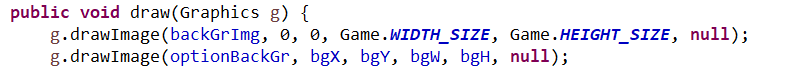
Ảnh có chứa đồ chơi

Mô tả được tạo tự động

Trước tiên chúng ta cần load các file ảnh, và vẽ nó lên màn hình, tuỳ thuộc vào các trạng thái game mà các menu sẽ hiện khác nhau.

Ảnh có chứa văn bản, bức thư

Mô tả được tạo tự động

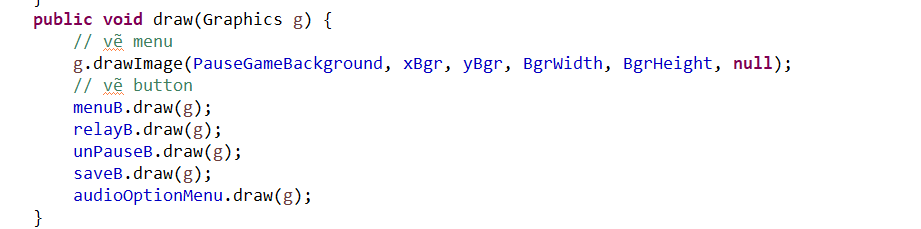


Tiếp theo tuỳ thuộc vào mỗi menu mà chúng ta sẽ phải tạo thêm cho cho các button khác nhau

Ảnh có chứa văn bản

Mô tả được tạo tự động

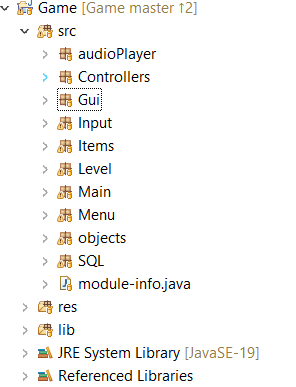
Tiếp theo là vẽ các button lên giao diện menu



Trên đây là một khuôn mẫu để code các menu trạng thái game. Các menu trạng thại khác trong game được code tương tự.

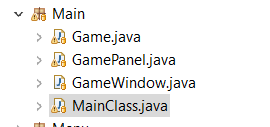
# **3. Mô hình các gói, các lớp của game**

Game được thiết kế chia nhỏ thành nhiều package khác nhau, mỗi package có các lớp với chức năng khác nhau.



\* Package chính là Main

Trong package Main có 4 lớp chính: đó là lớp game, lớp game window, lớp gamepanel, lớp main



- Lớp game là lớp để khởi tạo các trạng thái của game, vòng lặp cho game bao gồm vòng lặp về vẽ hình, vòng lặp update, khởi tạo trình điều khiển cho game, khởi tạo cơ sở dữ liệu.

- Trong lớp GamePanel: khởi tạo các trình xử lý sự kiện cho game như KeyListener, MouseListener

- Trong lớp GameWindow: khởi tạo khung hình Jframe cho game.

- Trong lớp Main: nơi khởi tạo game lớp chính của game.

\* Package Object: Dùng để tạo các đối tượng cho game, như nhân vật, kẻ thù

Ảnh có chứa văn bản

Mô tả được tạo tự động

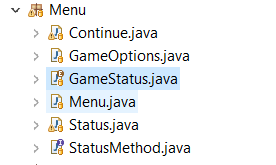
- Lớp Object: là lớp Abstracts có các thuộc tính chung cho nhân vật và kẻ thù như vị trí, các hộp sát thương, các trạng thái…

- Các lớp Player, Enemy: là các lớp được kế thừa từ lớp Object có các thuộc tính riêng của từng loại nhân vật sẽ khác với kẻ thù.

- Lớp Crabby là lớp kế thừa từ lớp Enemy: đây là lớp dùng để khởi tạo các kẻ thù cụ thể cho game vì game sẽ có nhiều loại kẻ thù.Mỗi kẻ thù sẽ có các thuộc tính riêng.

- Lớp EnemyManager là lớp quản lý kẻ thù, vì trong mỗi map thường có nhiều kẻ thù nên chúng ta cần quản lý list các kẻ thù trong map.

\* Package Menu: Đâu là package quản lý các trạng thái game: bao gồm các trạng thái Continue( đang chơi), Menu( Màn hình chính),GameOption, QuitGame.



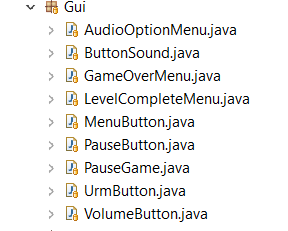
- Trong lớp Continue: Có các thuộc tính Nhân vậy, Quản lý kẻ thù, các trạng thái của game, các hàm bắt và xử lý sự kiện được truyền vào từ lớp gamePanel, có phương thức update trạng thái cho game, render…

- Trong lớp GameStatus đây là lớp được đinh nghĩa là Enum: bao gồm các trạng thái của game CONTINUE, MENU, OPTIONS, QUIT;

- Trong lớp Status: Đây là lớp để set các trạng thái của game khi game chuyển trạng thái từ trạng thái này sang trạng thái khác thông qua sự thay đổi logic của các biến trạng thái của game. Ví dụ trạng thái của Game đang là Continue. Khi đó có một biến Paused là biến true/ false để cho biến trạng thái của game đang là gì. Nếu game đang chạy thì biến này sẽ được mặc định là false. Khi ta thực hiện cho game tạm dừng bằng cách truyền vào sự kiện nhấn 1 phím xuống, sau khi nhân phím xuống thì phím sẽ truyền vào keyInput, khi đó trình quản lý GamePanel sẽ set lại trạng thái biến Paused= true; biến Paused bằng true -> trạng thái Pause game sẽ được thiết lập cho game, và màn hình Paused sẽ hiện lên. Nếu chúng ta nhấn nút tiếp tục trong màn hình Pause thì khi đó sự kiện kick vào button sẽ được thực hiện và biến paused sẽ đước set thành false, trạng thái continue sẽ được thiết lập và game lại quay trở lại trạng thái tiếp tục chơi. Có thể nói đây là cách thức mà các menu hoạt động cũng như cách thức di truyển của nhân vật trong game.

- Lớp StatusMethod: đây là interface dùng để định nghĩa các phương thức trừu tượng để ghi đè lại trong các trạng thái game. Các lớp continue, menu, gameOption sẽ kế thừa các interface này.

\* Package GUI: Đây là package quản lý các giao diện cho các menu trạng thái game, như pause game, gameOption, menu chính…. Trong package có các lớp giúp quản lý, khởi tạo giao diện, các button.



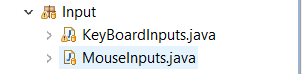
- Lớp Button Sound, volume, urmbutton, pausebutton: đều là các loại button được dùng để tạo các loại nút khác nhau để sử dụng lại nhiều làm. Trong các nút này đề có các MouseListener để lắng nghe sự kiện tương tác chuột.

- Lớp AudiooptionMenu: Khởi tạo giao diện cho gameOption, trong đây có khởi tạo các button điều khiển trạng thái của âm thanh, điều kiển volume ở trên.

- PauseGame: Khởi tạo giao diện Pause cho game, cũng có khởi tạo các button điển khiển trạng thái game, trạng thái âm thanh

- LevelCompleteMenu: Khởi tạo giao diện hoàn thành màn sau khi kết thúc màn, ở đây cũng khởi tạo các button điều khiển

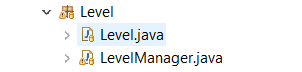
\* Package Inputs: Quản lý các trình nhận sự kiện để xử lý sự kiện cho game như KeyListener, MouseListener



- KeyBoardInput: Lớp này để thiết lập các trình xử lý sự kiện khi ta tương tác với bàn phím

- MouseInput: lớp này để thiết lập các trình xử lý sự kiên khi ta tương tác với chuột

\* Package Level: Đây là package để quản lý map



- Level: Quản lý từng level 1 bao gồn chỉ số định danh theo màu của các ô trong map, các ô định danh cho kẻ thù, định danh cho nhân vật, định danh cho item.

- Level Manager: Quản lý các level của game.

\* Package audioPlayer: quan lý âm thanh cho game

Ảnh có chứa văn bản

Mô tả được tạo tự động

- Lớp AudioPlayer: Lớp này dùng để quản lý các trình phát âm thanh cho game, thay đổi các hiệu ứng âm thanh, bật tắt âm thanh, điểu khiển âm lượng.

\* Package Controllers: chứa các lớp final tĩnh, các biến final tĩnh, các hàm check vị trí….

Ảnh có chứa văn bản

Mô tả được tạo tự động

- Controller: Chứa các biến final tĩnh cho game, như là các trạng thái của nhân vật, của kẻ thù, các kích thước của các button, các hình menu…

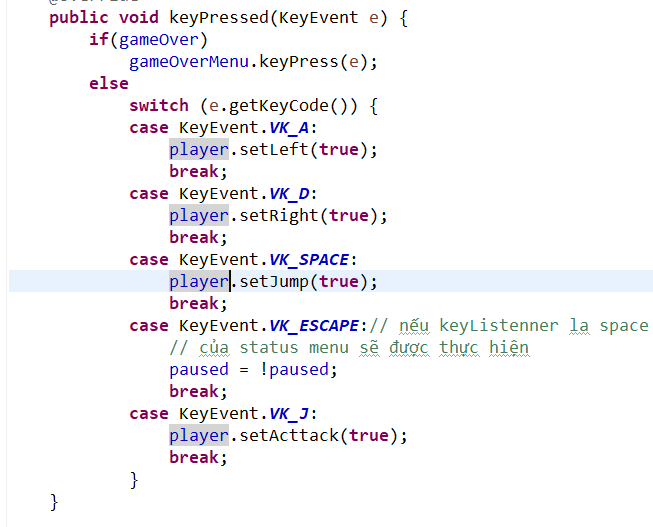
- Map: Quản lý danh sách các bức ảnh sử dụng cho game. Có các phương thức đọc file ảnh

- LogicChangePos: Chứa các phương thức tĩnh dùng để kiểm tra vị trí nhân vật trong map, kiểm tra vị trí nhân vật có trướng ngoại vật hay không, nhân vật có đứng trong các ô có các chướng ngoại vật được vẽ lên map không, vị trí tiếp theo mà nhân vật di truyển có chướng ngoại vật hay không, khởi tạo định danh theo giá trị màu cho các ô trong map.

# **4. Giải thích các logic game**

## **4.1 Logic di truyển của nhân vật và kẻ thù**

Trước hết ta xét di truyển của nhân vật: nhân vật có các biến trạng boolean chỉ trạng thái di truyển, tấn công của nhân vật đó là left, right, jump, attack. Biến này mặc định là false. Khi ra thực hiện nhấn các nút trên bàn phím thì game sẽ có trình nhận sự kiện sau khi nhận sự kiện xong thì nó sẽ thực hiện xử lý sự kiện, thay đổi tính logic của các biến trạng thái , các biến trạng thái chuyển về dạng true. Khi đó có một hàm cập nhật vị trí của nhân vật sẽ thực hiện kiểm tra điều kiện nếu như trạng thái left của nhân vật true thì sẽ thực hiện thay đổi tốc dộ di truyển của nhân vật. tương tự đối với các trạng thái khác bao gồm cả nhảy, và tấn công.

 Ảnh có chứa văn bản

Mô tả được tạo tự động

Tuy nhiên có một vấn đề là trong game thì nhân vật vẫn có thể di truyển ở khắp mọi nơi trên bản đồ bao gồm đi lên cả các chướng ngoại vật hay là cả không khí, như thế chúng ta cần phải trải map ra thành map dọc. Đó là thiết lập cho game một trọng lực để nhân vật luôn rơi xuống dưới của map, cũng như là cần phải xét các vị trí mà nhân vật có thể đi tới.

Để thực hiện được điều này ta cần phải thêm cho nhân vật, cũng như kẻ thù các hộp bao quanh chứa toàn bộ nhân vật và kẻ thù. Các hộp này gọi là các hộp hitBox.

Để nhân vật có thể rơi trên map chúng ta cần phải xét xem dưới chân nhân vật có chướng ngoại vật nào không. Để xét như vậy thì chúng ta cần phải xét giao giữa hộp nhân vật và ô chứa nhân vật xem ô chứa nhân vật có chướng ngoại vật nào không. Như ta đã giải thích ở phần vẽ map thì mỗi ô trong map sẽ được định danh bởi một giá trị của giá trị màu. Ta cần biết được giá trị màu ở ô đó là bao nhiêu, nếu như giá trị màu ở ô đó nằm từ 0-48 và khác 11 thì ô đó có chướng ngoại vật tức là không thể đi được.

Ảnh có chứa văn bản

Mô tả được tạo tự động

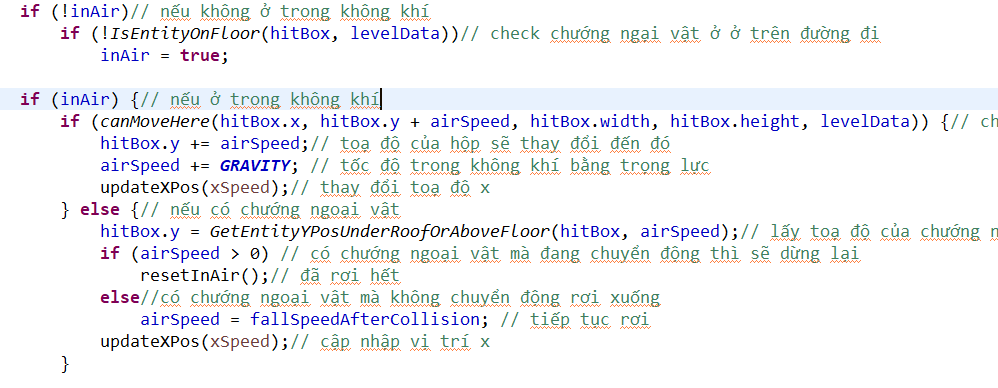
Đây chính là phương thức để kiểm tra xem ô đó có chướng ngoại vật hay không nếu giá trị là true tức là ô đó không có trướng ngoại vật false tức là có.

Ảnh có chứa văn bản

Mô tả được tạo tự động

Đây là phương thức để lấy ô đang đứng xong gọi đến hàm kiểm tra chướng ngoại vật của ô hiện tại

Tiếp theo chúng ta cần kiểm tra xem nhận vật có đang đứng trên chướng ngoại vật không nếu không thì tiếp tục rơi xuống nếu có thì cập nhật vị trí nhân vật.

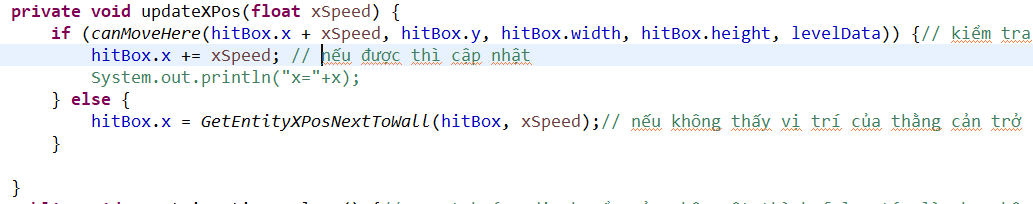


Tiếp theo chúng ta cần xét kiểm tra xem các vị trí mà nhân vật có thể di truyển trong map. Ta cũng cần xét xem ô tiếp theo mà nhân vật di truyển đến có chướng ngoại vật không nếu có thì không được di truyển nếu không thì tiếp tục được di truyển

Ảnh có chứa văn bản

Mô tả được tạo tự động

Trên đây là hàm kiểm tra xem vị trị tiếp theo nhân vật có di truyển được không

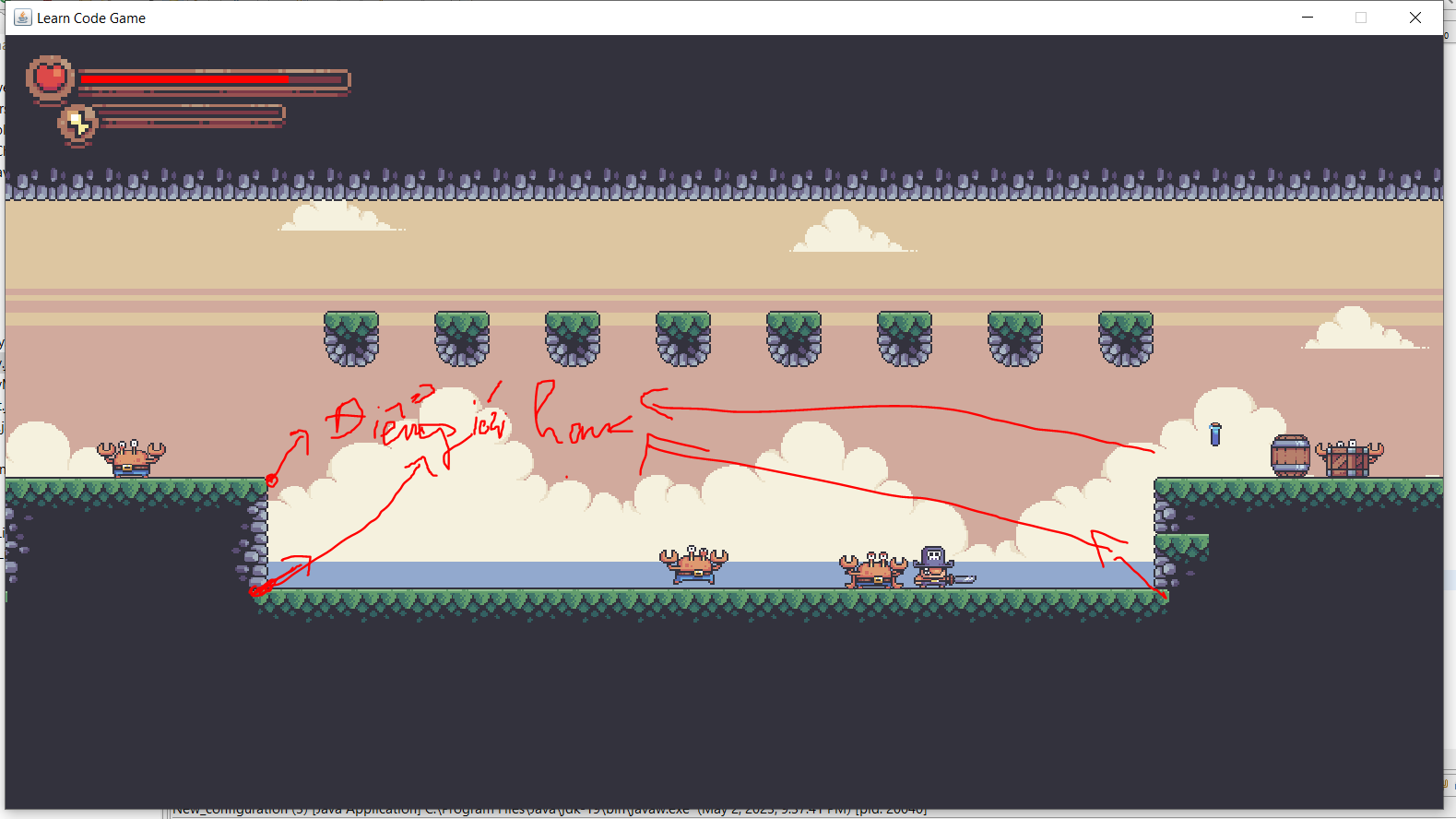


\* Logic di truyển của kẻ thù

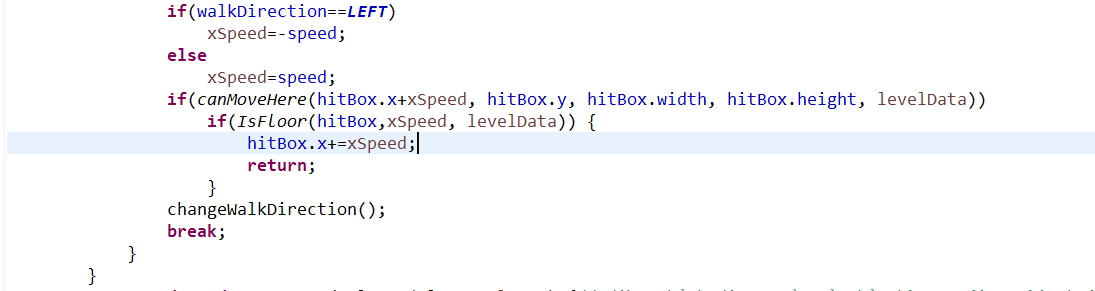
Đối với kẻ thù cũng có các logic di truyển giống như nhân vật. Tuy nhiên chúng ta không thiết lập các điều khiển logic di truyển của kẻ thù như nhân vật mà logic di truyển của kẻ thù được tạo ra lặp đi lặp lại, không chịu sự điều khiển của người chơi.

Ta sẽ code một số hàm thiết lập di truyển cho kẻ thù:

- Khi kẻ thù chưa nhìn thấy nhân vật thì sẽ di truyển theo một hướng cho đến khi đi đến điểm giới hạn trái hoặc phải của map( điểm này là điểm mà tại đó có chướng ngoại vật)



Khi kẻ thù đi đến điểm giới hạn thì biến hướng đi của kẻ thù sẽ thay đổi



Ảnh có chứa văn bản

Mô tả được tạo tự động

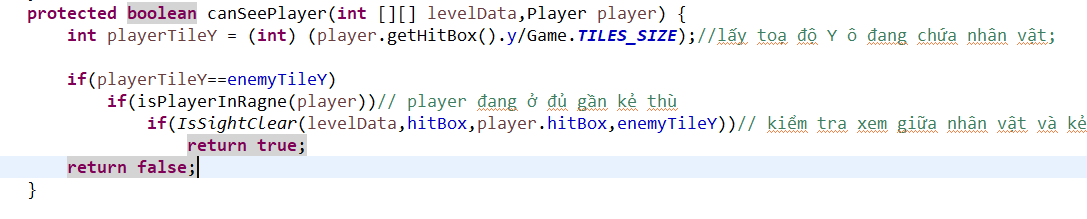
Tiếp theo chúng ta cần phải thiết lập các logic di truyển cho kẻ thù khi có nhân vật

- Trước tiên ta cần phải kiểm tra xem nhân vật có nằm trong tầm ngắm của kẻ thù không nếu nhân vật đang nằm trong tầm ngắm của kẻ thù thì kẻ thù sẽ luôn đuổi theo nhân vật.

Ảnh có chứa văn bản

Mô tả được tạo tự động

Khoảng cách kẻ thù và nhân vật



Check kẻ thù đang ở gần nhân vật

Ảnh có chứa văn bản

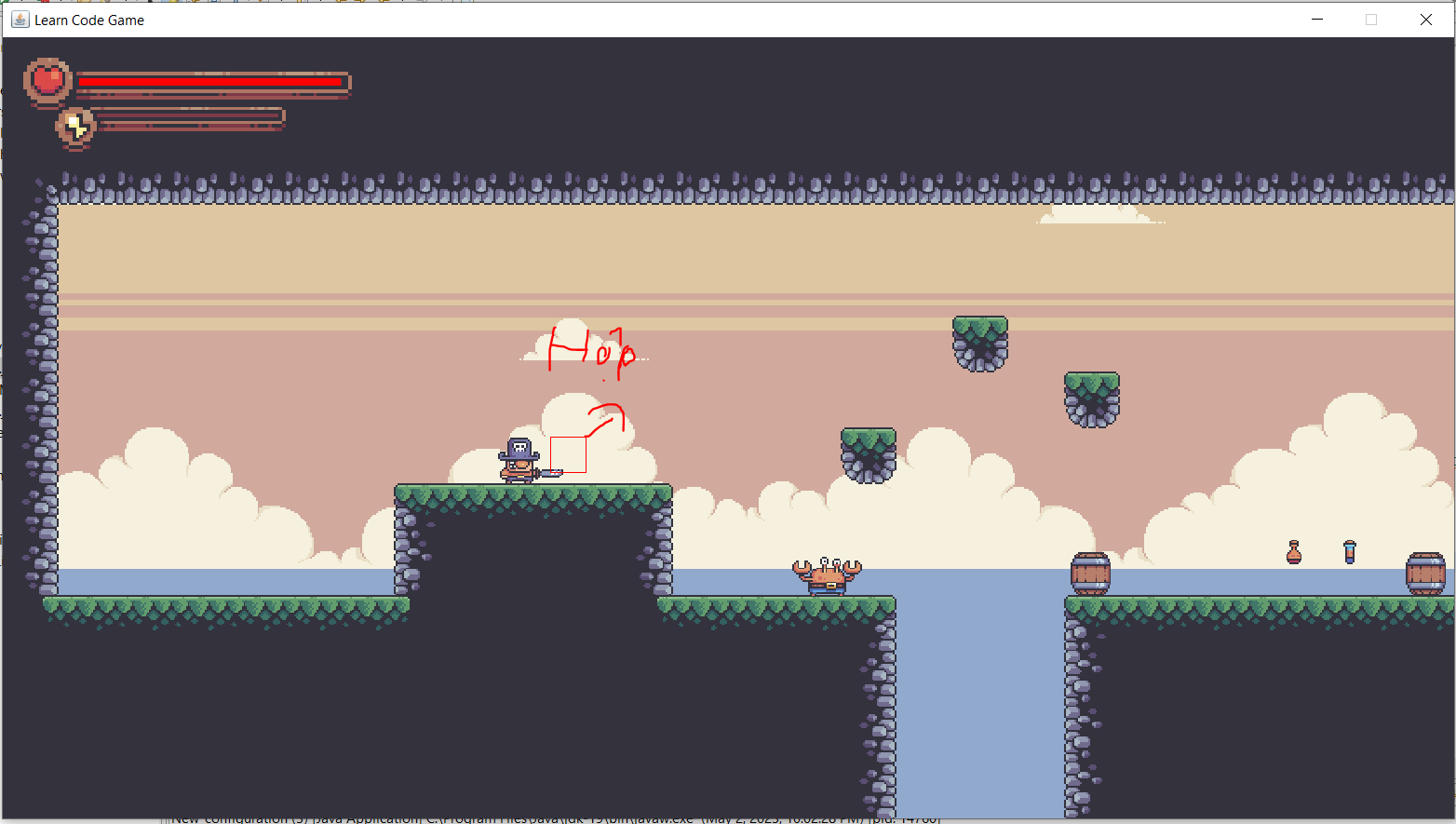
Mô tả được tạo tự động

Kẻ thù đuổi theo nhân vật

## **4.2. Logic gây sát thương của nhân vật và kẻ thù**

\* Nhân vật gây sát thương cho kẻ thù

Ta tạo cho nhân vật một vùng, vùng này có tác dụng xét va chạm với kẻ thù



Khi kẻ thù đi vào vùng không gian của hộp và khi đó trạng thái của nhân vật là attack thì kẻ thù sẽ bị trừ máu.

Ảnh có chứa văn bản

Mô tả được tạo tự động

Ảnh có chứa văn bản

Mô tả được tạo tự động

\* Kẻ thù gây sát thương cho nhân vật:

Như mục trên em đã đề cập đến quá trình di truyển của kẻ thù khi nhân vật đi vào tầm ngắm của nhân vật.

Tiếp theo em sẽ mô tả lại quá trình tấn công của kẻ thù

Trước tiên ta sẽ kiểm tra xem nhân vật có đang trong phạm vi tấn công của kẻ thù không

Ảnh có chứa văn bản

Mô tả được tạo tự động

Tiếp theo ta sẽ kiểm tra va chạm giữa hộp tấn công của kẻ thù và hộp nhân vật nếu 2 hộp này giao nhau thì máu kẻ thù bị giảm

Ảnh có chứa văn bản

Mô tả được tạo tự động

Như vậy tổng kêt phần 4.2 em đã giải thích về logic di truyển và logic tấn công của kẻ thù và nhân vật. Để đảm bảo dung lượng bài báo cáo nên em đã tóm lược ngắn gọn phần giải thích ngắn nhất có thể.

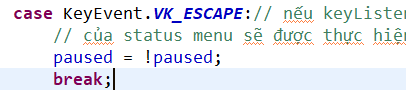
## **4.3. Logic chuyển trạng thái của game**

Trong lớp Continue tức là lớp quản lý tất cả mọi đối tượng hiện thị hiện tại trong game. Trong lớp này sẽ có các biến boolean để chuyển các trạng thái menu cho game

Ảnh có chứa văn bản

Mô tả được tạo tự động

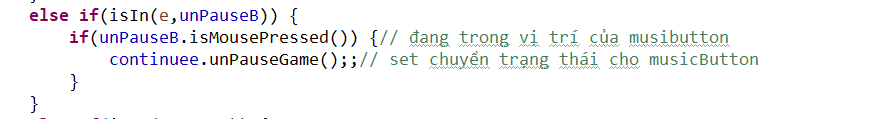
Khi game đang chạy thì biện pause sẽ là false. Khi ta thực hiện nhận phím tạm dừng cho game truyền vào phím escape ta đã thiết lập xử lý sự kiện khi nhấn phím escape khi nhấn vào thì pause sẽ chuyển thành true, trong lớp Continue có một hàm check điều kiện nếu pause =true thì sẽ truyển trạng thái của game đang tự status continue thành pause game, khi đó menu pause game sẽ hiện lên. Và tất cả các trạng thái của continue sẽ dừng hết lại



Ảnh có chứa văn bản

Mô tả được tạo tự động

Khi ta nhấn tiếp tục chơi thì thiết lập lại giá trị false cho biến pause sau đó check điều kiền pause trên nếu không đúng thì sẽ không thiết lập trạng thái Pause nữa và quay về trạng thái continue.



Đối với các trạng thái khác logic cũng tương tự như Pause Game.

*Kết luận phần 4***:** Sau phần 4 em cũng đã tóm lược lại logic của game từ cách nhân vật di truyển, các nhân vật và kẻ thù tấn công, cách chuyển qua lại giữa các menu trạng thái game.

# **5. Kết Luận**

Sau quá trình tự tìm hiểu trên internet, kết hợp với kiến thức thầy truyền đạt trên lớp em đã code ra được game cho bản thân. Em đã hiểu thêm về quá trình xây dựng các lớp cho một project, hiểu về cách hoạt động, các bước để phát triển ra một game. Sau này nếu có điều kiện về thời gian chắc chắc em sẽ tiếp tục hoàn thiện sản phẩm cũng như là có thêm một số game khác. Trên đây là báo cáo về bài tập lớn của em. Nếu còn nhiều thiếu sót, em rất mong nhận được sự góp ý của thầy để sản phẩm của em thêm hoàn thiện cũng như là em nhận được thêm những bài học về các dự án lần sau. Em trân thành cảm ơn thầy trong suốt học phần đã truyền đạt nhiều kiến thức cho em.

# **6. Tài liệu tham khảo**

Toàn bộ slide của thầy.

https://uet.vnu.edu.vn/~chauttm/e-books/OOP\_2013.pdf

https://chat.openai.com/

https://www.youtube.com/watch?v=6\_N8QZ47toY&list=PL4rzdwizLaxYmltJQRjq18a9gsSyEQQ-0