**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI**

**KHOA CÔNG NGHỆ THÔNG TIN**

---o0o---

**BÁO CÁO BÀI TẬP LỚN MÔN CÔNG NGHỆ JAVA**

**ĐỀ TÀI: LÀM GAME 2D KHỦNG LONG**



Sinh viên thực hiện:

Nguyễn Nhật Trường (MSSV: 211240507)

Nguyễn Đức Nghĩa (MSSV: 211202183)

*Hà Nội, tháng 4 năm 2023*

**LỜI MỞ ĐẦU**

Trong thời đại công nghệ hiện nay, game đang trở thành một lĩnh vực phát triển nhanh chóng và thu hút sự quan tâm của rất nhiều người. Game không chỉ mang tính giải trí mà còn là một công cụ để giúp con người học hỏi và giải quyết các vấn đề thực tế.

Trong đề tài này, báo cáo sẽ tập trung vào việc phát triển một game 2D bằng ngôn ngữ lập trình Java. Java là một ngôn ngữ lập trình đa nền tảng, cho phép chúng ta phát triển các ứng dụng trên nhiều hệ điều hành và các thiết bị khác nhau. Với sự phổ biến của Java, việc phát triển game 2D bằng ngôn ngữ này là một lựa chọn lý tưởng.

Bài tập lớn này sẽ tìm hiểu về các khái niệm cơ bản trong phát triển game 2D, cách thiết kế và xây dựng game. Đồng thời, chúng ta cũng sẽ giải quyết các thách thức trong quá trình phát triển game như xử lý các nhân vật, chướng ngại vật…

Hy vọng rằng báo cáo này sẽ giúp mọi người hiểu rõ hơn về quá trình phát triển game 2D bằng ngôn ngữ Java và các kỹ thuật để xây dựng một game chất lượng cao.

MỤC LỤC

[PHẦN I: GIỚI THIỆU ĐỀ TÀI 4](#_Toc133942624)

[1. Mô tả yêu cầu bài toán: 4](#_Toc133942625)

[2. Biểu đồ USE CASE: 4](#_Toc133942626)

[3. Đặc tả use case: 4](#_Toc133942627)

[PHẦN 2: THIẾT KẾ CHI TIẾT 4](#_Toc133942628)

[1. Thiết kế lớp 4](#_Toc133942629)

[2. Thiết kế chi tiết lớp: 5](#_Toc133942630)

[PHẦN 3 : CHƯƠNG TRÌNH MINH HỌA 12](#_Toc133942631)

[Xây dựng chương trình: 12](#_Toc133942632)

[Kết quả chương trình: 12](#_Toc133942633)

[Phần 4: Hướng dẫn cài đặt: 13](#_Toc133942634)

# PHẦN I: GIỚI THIỆU ĐỀ TÀI

## Mô tả yêu cầu bài toán:

* Điều khiển con khủng long di chuyển và nhảy trên môi trường chơi game.
* Tránh các chướng ngại vật.
* Có hệ thống điểm số để đánh giá kết quả của người chơi.
* Thể hiện thông báo khi người chơi đã bị kết thúc game.
* Cho phép người chơi bắt đầu lại game hoặc thoát khỏi game.
* Trò chơi có âm thanh khi người dùng nhảy hoặc gặp chướng ngại vật.

## Biểu đồ USE CASE:

* Tác nhân: Người chơi

USE case:

* Start Game
* Move
* Dead

## Đặc tả use case:

* Start Game:

+ Mô tả: Người dùng sử dụng use này để bắt đầu gane

+ Dòng sự kiện: Người dùng ấn phím Space bắt đầu

* Move:

+ Mô tả: Người dùng sử dụng use này để di chuyển

+ Dòng sự kiện: Người dùng ấn phím Space để nhảy lên, phím mũi tên xuống để cúi người

* Dead:

+ Mô tả: Người dùng sử dụng use này để kết thúc game

+ Dòng sự kiện: Người dùng chạm vào chướng ngại vật

# PHẦN 2: THIẾT KẾ CHI TIẾT

## 1. Thiết kế lớp

Các lớp bao gồm:

Package gameobject:

* class Bird
* class Cactus
* class Clouds
* class EnemiesManager
* class Enemy
* class Land
* class MainCharacter

package gameobject:

* class GameScreen
* class GameWindow

package util:

* class Animation
* class Resource

## 2. Thiết kế chi tiết lớp:

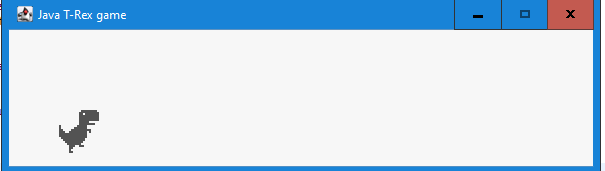
* Lớp Bird kế thừa từ lớp Enemy:
* Thuộc tính:
* private Random rand: đối tượng Random được sử dụng để sinh ra vị trí ngẫu nhiên của chim.
* private int posY: vị trí y của chim.
* private int posX: vị trí x của chim.
* private int width: chiều rộng của chim.
* private int height: chiều cao của chim.
* private BufferedImage image: ảnh của chim.
* private MainCharacter mainCharacter: đối tượng MainCharacter mà chim đang tấn công.
* private Rectangle rectBound: đối tượng Rectangle để xác định ranh giới của chim.
* Phương thức:
* public Bird(MainCharacter mainCharacter, int posX, int width, int height, BufferedImage image): phương thức khởi tạo cho đối tượng Bird với các tham số được truyền vào bao gồm đối tượng MainCharacter, vị trí x, chiều rộng, chiều cao và ảnh.
* public void update(): phương thức cập nhật vị trí của chim dựa trên tốc độ của MainCharacter.
* public void draw(Graphics g): phương thức vẽ chim lên đối tượng Graphics và vẽ ranh giới của nó.
* public Rectangle getBound(): phương thức trả về đối tượng Rectangle để xác định ranh giới của chim.
* public boolean isOutOfScreen(): phương thức kiểm tra xem chim đã di chuyển ra khỏi màn hình hay chưa. Nếu vị trí x của chim nhỏ hơn chiều rộng của ảnh của nó, phương thức trả về true. Ngược lại, nó trả về false.
* Lớp Cactus
* Thuộc tính:
* posX: Vị trí x của cây xương rồng trên màn hình.
* width: Chiều rộng của cây xương rồng.
* height: Chiều cao của cây xương rồng.
* image: Hình ảnh của cây xương rồng.
* mainCharacter: Tham chiếu đến đối tượng MainCharacter (kẻ chơi) để có thể xác định tốc độ di chuyển của đối tượng Cactus.
* rectBound: Đối tượng Rectangle để xác định vùng giới hạn của đối tượng Cactus.
* Phương thức:
* Cactus(MainCharacter mainCharacter, int posX, int width, int height, BufferedImage image): Constructor của lớp Cactus để khởi tạo đối tượng Cactus với các thông số truyền vào như vị trí, kích thước, hình ảnh và tham chiếu đến đối tượng MainCharacter.
* update(): Phương thức cập nhật vị trí của đối tượng Cactus trên màn hình dựa trên tốc độ của MainCharacter.
* draw(Graphics g): Phương thức để vẽ đối tượng Cactus lên màn hình và vẽ ra đường biên (border) cho vùng giới hạn của đối tượng Cactus.
* getBound(): Phương thức trả về đối tượng Rectangle, đại diện cho vùng giới hạn của đối tượng Cactus.
* isOutOfScreen(): Phương thức kiểm tra xem đối tượng Cactus có nằm ngoài màn hình không, dựa trên vị trí của đối tượng và kích thước của hình ảnh của nó. Nếu đối tượng đã nằm ngoài màn hình, phương thức trả về true, ngược lại trả về false.
* Các thuộc tính và phương thức của lớp Clouds:
* Thuộc tính:
* listCloud: danh sách chứa các hình ảnh mây.
* cloud: hình ảnh mây được tải từ tệp ảnh.
* mainCharacter: tham chiếu đến đối tượng MainCharacter.
* Phương thức:
* Constructor Clouds(int width, MainCharacter mainCharacter): tạo đối tượng Clouds với chiều rộng màn hình và đối tượng MainCharacter.
* void update(): cập nhật vị trí của các hình ảnh mây, di chuyển theo vận tốc của đối tượng MainCharacter.
* void draw(Graphics g): vẽ các hình ảnh mây lên màn hình game.
* Lớp ImageCloud: lớp nhỏ bên trong lớp Clouds, chứa các thuộc tính của từng hình ảnh mây như vị trí posX và posY.
* Trong lớp EnemiesManager, chúng ta có:
* Thuộc tính:
* cactus1: lưu trữ hình ảnh của cây xương rồng loại 1.
* cactus2: lưu trữ hình ảnh của cây xương rồng loại 2.
* bird: lưu trữ hình ảnh của chim.
* rand: đối tượng Random để tạo ngẫu nhiên loại kẻ thù.
* enemies: danh sách các đối tượng Enemy hiện tại.
* mainCharacter: đối tượng MainCharacter để xử lý va chạm và tính điểm.
* Phương thức:
* EnemiesManager(MainCharacter mainCharacter): constructor, khởi tạo các thuộc tính và danh sách enemies, đồng thời thêm 2 đối tượng kẻ thù đầu tiên bằng phương thức createEnemies().
* createEnemies(): trả về một danh sách gồm 2 đối tượng kẻ thù được tạo ngẫu nhiên.
* update(): cập nhật trạng thái của danh sách enemies và kiểm tra xem đối tượng kẻ thù cuối cùng có nằm ngoài màn hình hay không. Nếu có, thêm 2 đối tượng kẻ thù mới bằng phương thức createEnemies() và xóa danh sách enemies hiện tại.
* draw(Graphics g): vẽ danh sách enemies lên màn hình.
* isCollision(): kiểm tra xem có va chạm giữa mainCharacter và các đối tượng kẻ thù hay không. Nếu có, trả về true, ngược lại trả về false.
* reset(): xóa danh sách enemies hiện tại và thêm 2 đối tượng kẻ thù mới bằng phương thức createEnemies().
* Lớp Enemy:
* Các phương thức:
* update()
* draw(Graphics g)
* getBound()
* isOutOfScreen()
* Lớp Land:
* Các thuộc tính:
* private List<ImageLand> listLand;
* private BufferedImage land1;
* private BufferedImage land2;
* private BufferedImage land3;
* private MainCharacter mainCharacter;
* Các phương thức:
* public Land(int width, MainCharacter mainCharacter)
* public void update()
* private void setImageLand(ImageLand imgLand)
* public void draw(Graphics g)
* private int getTypeOfLand()
* private class ImageLand
* Lớp MainCharacter
* Các thuộc tính:
* posY
* posX
* speedX
* speedY
* rectBound
* score
* state
* normalRunAnim
* jumping
* downRunAnim
* deathImage
* jumpSound
* deadSound
* scoreUpSound
* Các phương thức:
* getSpeedX()
* setSpeedX()
* draw()
* update()
* jump()
* down()
* getBound()
* dead()
* reset()
* playDeadSound()
* upScore()
* Tên class: GameScreen
* Phương thức:
* startGame(): bắt đầu chạy thread.
* gameUpdate(): cập nhật trạng thái của game.
* paint(Graphics g): vẽ các đối tượng trong game.
* keyPressed(KeyEvent e): xử lý sự kiện nhấn phím.
* keyReleased(KeyEvent e): xử lý sự kiện thả phím.
* keyTyped(KeyEvent e): xử lý sự kiện gõ phím.
* run(): chạy game.
* Thuộc tính:
* land: đối tượng Land.
* mainCharacter: đối tượng MainCharacter.
* enemiesManager: đối tượng EnemiesManager.
* clouds: đối tượng Clouds.
* thread: đối tượng Thread.
* isKeyPressed: biến boolean xác định có phím được nhấn hay không.
* gameState: trạng thái của game (bắt đầu, đang chơi, kết thúc).
* replayButtonImage: ảnh nút chơi lại.
* gameOverButtonImage: ảnh chữ game over.
* Tên class: GameWindow
* Thuộc tính:
* SCREEN\_WIDTH (final int)
* gameScreen (GameScreen)
* Phương thức:
* GameWindow() (constructor)
* startGame()
* main(String[])
* Tên class: Animation
* Thuộc tính:
* list: ArrayList<BufferedImage>: danh sách các ảnh được thêm vào để tạo animation
* deltaTime: long: khoảng thời gian giữa mỗi frame trong animation
* currentFrame: int: chỉ số của frame hiện tại đang được hiển thị
* previousTime: long: thời điểm trước đó của frame hiện tại
* Phương thức:
* Animation(int deltaTime): khởi tạo một đối tượng Animation với đối số là deltaTime
* void updateFrame(): cập nhật frame hiện tại để hiển thị
* void addFrame(BufferedImage image): thêm một frame vào danh sách các frame trong animation
* BufferedImage getFrame(): lấy ra frame hiện tại để hiển thị.
* Tên class: Resource
* Các phương thức:
* getResouceImage(String path): trả về một đối tượng BufferedImage dựa trên đường dẫn của file ảnh được truyền vào dưới dạng String. Nếu không tìm thấy file ảnh tương ứng hoặc có lỗi khi đọc file, phương thức sẽ trả về giá trị null.

# PHẦN 3 : CHƯƠNG TRÌNH MINH HỌA

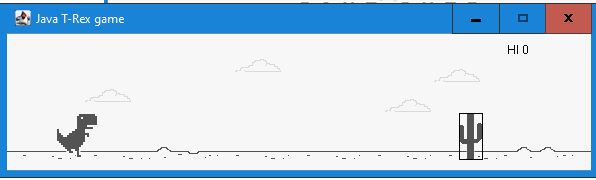
## Xây dựng chương trình:

* Công nghệ sử dụng: JAVA
* Môi trường lập trình: Eclipse IDE

## Kết quả chương trình:



Khi bắt đầu vào game



Khi đang chơi



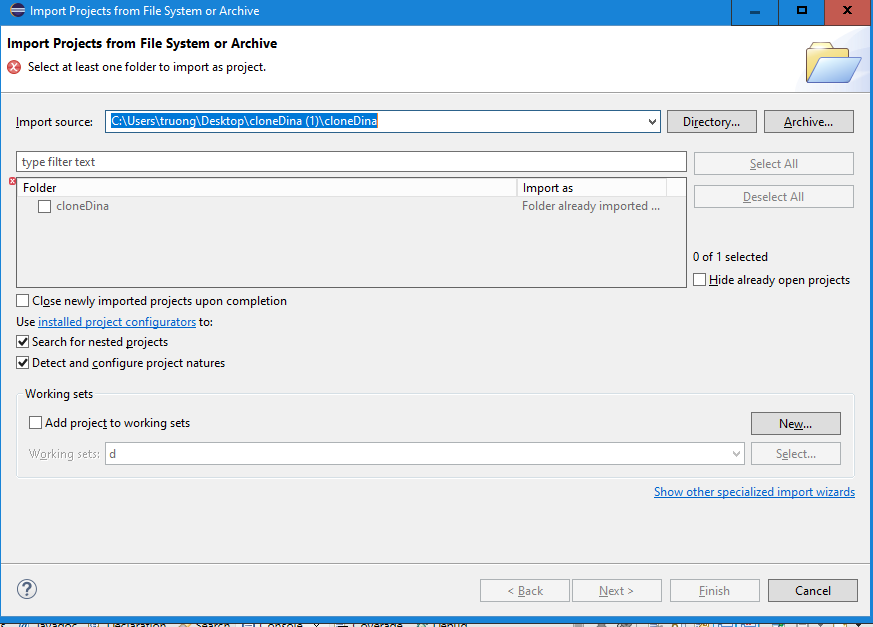
Khi kết thúc

# Phần 4: Hướng dẫn cài đặt:

B1: Cài đặt Eclipse IDE(dowload tại <https://eclipse.org/downloads/>)

B2:Vào File, Chọn Open Project From File System

B3 : Ấn vào Directory để chọn thư mục chứa project



B4: Ấn Finish

B5: Ấn vào project vừa thêm và ấn F11 để chạy

