The Farmer and The Viper Documentation

Created by

Punyaphat Surakiatkamjorn 6432106821

Thanat Wongsamut 6432067021

2110215 Programming Methodology

Semester 1 Year 2022

Chulalongkorn University

The Farmer and The Viper

1. Game Overview

The Farmer's farm is invaded by large herds of vipers. Instead of allowing the vipers to live in his field as they please. Instead, the farmer chose to fight for their honor and dignity. and want to destroy the myth that at the end of The Farmer and The Viper story must end by the farmer is bitten by the viper.

1.1. Gameplay

The Farmer and The Viper is a 2D shooting game inspired by 2D shooting game of our childhood. The objective of the game is to survive until all the vipers are gone.

1.1.1. Game Component

1.1.1.1. Player

- At the start, the player stands in the middle of floor. The player will have 100 HP

1.1.1.2. Floor

- Floor shows the area that player and viper can move around

1.1.1.3. Enemy

1.1.1.3.1. Viper

- Each Viper has 50 HP and can bite player with 10 damage. At the start, 3 Vipers will spawn at the edge of floor randomly. The vipers will respawn after they get killed immediately until the King Viper die. Before 5 minutes, the number of them increases 1 every 20 seconds.

1.1.1.3.2. King Viper

- King Viper will be summoned at 5 minutes with 10000 HP. Its movement is very slow but it can bite player and cause 90 damage. When it dies, the Vipers will stop respawning after they get killed.

1.1.1.4. Weapon

1.1.1.4.1. Pistol

- Pistol can shoot with 20 damage with fire rate 3.33 shoots per second. This gun can be used from the beginning of the game.

1.1.1.4.2. Rifle

- Rifle can shoot with 25 damage with fire rate 3.33 shoots per second. Player can use this gun after 1 minute after game started.

1.1.1.4.3. Shotgun

- The shotgun can fire 3 spreading bullets per shoot, each dealing 20 damage with fire rate 2 shoots per second. Player can use this gun after 1 minute after game started.

1.1.1.4.4. Machinegun

- Machinegun can shoot with 5 damage with fire rate 15 shoots per second. Player can use this gun after 3 minutes after game started.

1.1.1.4.5. Laser gun

- Laser gun can shoot with 40 damage with fire rate 1.25 shoots per second. This gun bullet can penetrate enemy and all enemy that are penetrated will receive damage. Player can use this gun after 3 minutes after game started.

1.1.1.4.6. Flamethrower

- Flamethrower can shoot with 2 damage with fire rate 6.66 shoots per second. This gun bullet can penetrate enemy and all enemy that are penetrated will receive damage every 20 millisecond until the bullet passes it. Player can use this gun after 3 minutes after game started.

1.1.1.5. Gun Inventory

- Gun inventory shows all guns that player can use. The available gun that player can use at the current time will show a non-transparency image of that gun. For the gun that does not allow player use, it will show a transparency image of that gun

1.1.1.6. Timer

- Timer shows the time played since the game started.

1.1.1.7. Score

- At the start, Score will be set to 0 and score multiplier will be set to 1. Player can get score when player kills enemy. Each viper killed will give a 10 score (Killing King Viper will give 2000 score) multiply by score multiplier. To increase the score multiplier by 1 (the score multiplier cannot exceed 20), player must kill 10 vipers without taking any damage. But if player take damage, score multiplier will be reset to 1.

1.1.2. Game Flow

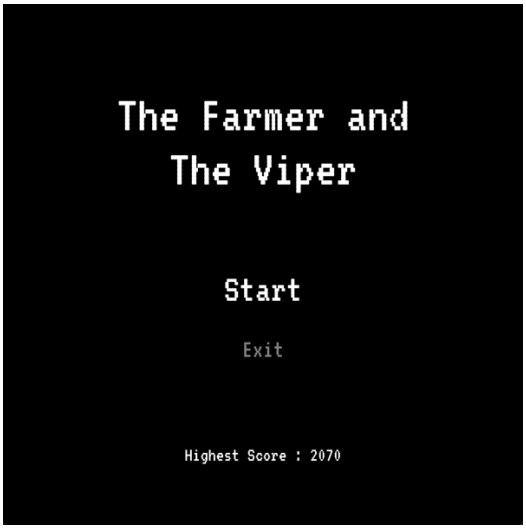
At the start, the player stands in the middle of floor with 3 vipers that are spawned at the edge of floor randomly. At the bottom of the window, you will see the gun list. The available gun you can use will show you a non-transparency image. The vipers will respawn after they get killed immediately. And the amount of them will increase by 1 every 20 seconds. If player can kill 5 vipers without taking any damage, player heal for 10 HP. The game is divided into 5 game periods.

- Before 1 minute, player is allowed to use only pistol.
- After 1 minute, player can now use shotgun and rifle.
- After 3 minutes, player can use all of guns.
- After 5 minutes, the King Viper is summoned, and the number of vipers no longer increases. But they will respawn after they are killed as before.
- And after the King Viper die, the vipers will not respawn.

If player can kill all vipers including The King Viper, game end and player win.
But if player takes damage until 0 HP, game over.

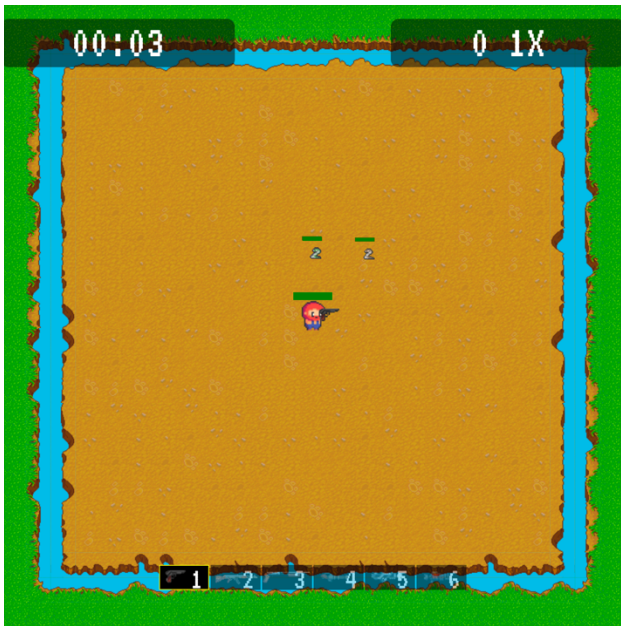
1.2. Example

1.2.1. Starting menu



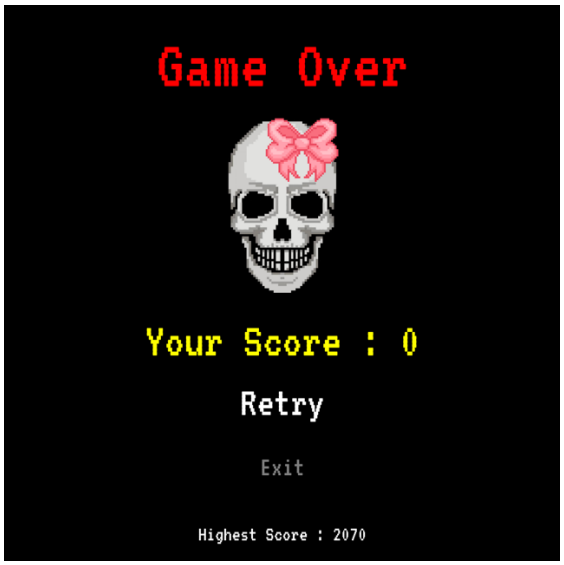
On this page, the name of the game will be displayed with start Button, exit Button and High Score. If we press start button, it will enter the game also if pressed exit button, it will close this window.

1.2.2. Playing screen



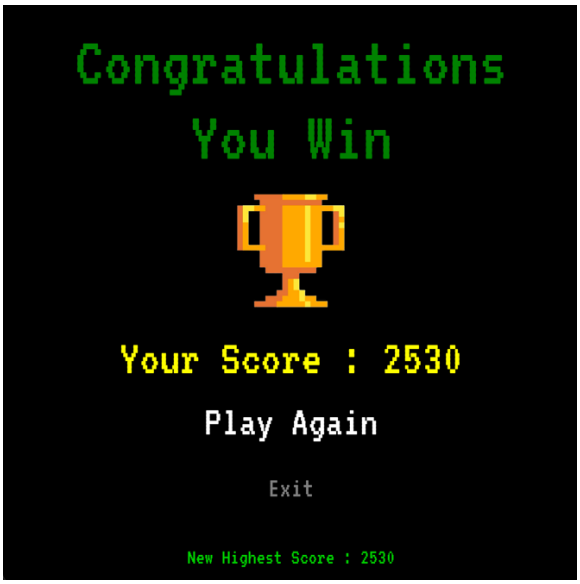
At the start, the player stands in the middle of floor with 3 vipers. Player must kill these vipers for surviving and collecting score

1.2.3. Lose screen



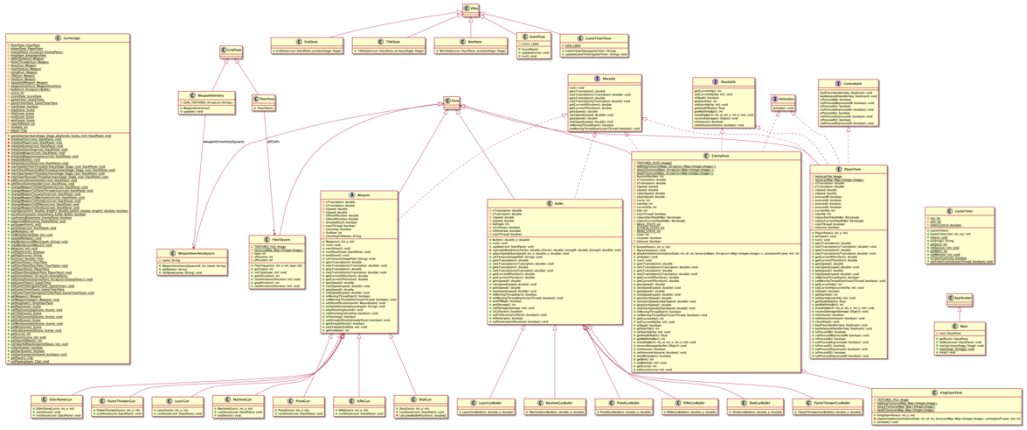
If player Hp is down to 0, the ending scene with the final score will show.

1.2.4. Won screen



If player can kill vipers until there is no viper left, won scene with final score will show.

2. Implementation Details:



* Noted that Access Modifier Notations are listed below

+ (public), # (protected), - (private), <u>underlined</u> (static), *Italic (abstract),* ALL_CAPS (final)

2.1. package application

2.1.1. Class Main extends Application

2.1.1.1. Fields

- StackPane root	Root Pane

2.1.1.2. Methods

+ start(Stage primaryStage)	- Initialize field - Show PrimaryStage
+ void main(String[] args)	Main application
+ void stop()	Close program

2.1.2. Class TitleState extends Vbox

2.1.2.1. Constructors

+ TitleState(StackPane root, Stage primaryStage)	Initialize title state

2.1.3. Class EndState extends Vbox

2.1.3.1. Constructors

+ EndState(StackPane root, Stage primaryStage)	Initialize end state

2.1.4. Class WonState extends Vbox

2.1.4.1. Constructors

+ WonState(StackPane root, Stage primaryStage)	Initialize won state

2.2. package gui.enemy

2.2.1. Class EnemyPane extends Pane implements Movable, Deadable, Animation

2.2.1.1. Fields

- Image[] TEXTURES_FILES	Store texture file
- ArrayList<Map<Integer, Image>> walkingTexturesMaps	Store walking animation
- ArrayList<Map<Integer, Image>> attackTexturesMaps	Store attacking animation
- ArrayList<Map<Integer, Image>> deathTexturesMaps	Store death animation
- int textureNumber	texture number of enemy
- double xTranslation	X Position of enemy
- double yTranslation	Y Position of enemy
- double xSpeed	X Current Speed of enemy
- double ySpeed	Y Current Speed of enemy
- double xStartSpeed	X Starting Speed of enemy
- double yStartSpeed	Y Starting Speed of enemy
- int score	Score that enemy when it dies
- int startHp	Start Hp of enemy
- int currentHp	Current Hp of enemy
- int bite	Bite attack of enemy
- boolean startThread	True when need enemy move False when need enemy stop
- Rectangle statusStartHealthBar	Show damage that enemy take
- Rectangle statusCurrentHealthBar	Show enemy's current Hp
# int WALK_STATE	Walk state
# int ATTACK_STATE	Attack state
# int DEAD_STATE	Dead state
# int state	Current state
# boolean respawn	True when enemy need respawns
- boolean immune	True when enemy can't take damage

2.2.1.2. Constructors

+ EnemyPane(int x, int y)	Construct Viper with the given position and initialize texture number, texture animation, score, Start Hp, Current Hp, Hp Bar, speed, respawn, bite damage, and size of enemy

2.2.1.3. Methods

# void flipAnimation()	Flip animation when action (biting,0 moving) direction opposite to the current animation
- int playAnimation(int animationState, int id, ArrayList<Map<Integer, Image>> texturesMaps, int animationFrame)	Reset id if state has transited and show animation (id)th frame via the current state

+ void animate()	Play animation and start that thread and set respawn to true if animation is end
+ void run()	Create Thread for calculating x and y speed for walking to player and updating enemy position in screen and start that thread
+ double getxTranslation()	Return x position
+ void setxTranslation(double xTranslation)	Set x position
+ double getyTranslation()	Return y position
+ void setyTranslation(double yTranslation)	Set y position
+ double getCurrentXPosition()	Return x Position on screen
+ double getCurrentYPosition()	Return y Position on screen
+ double getxSpeed()	Return x Current Speed
+ void setxSpeed(double speed)	Set x Current Speed
+ double getySpeed()	Return y Current Speed
+ void setySpeed(double speed)	Set y Current Speed
+ double getxStartSpeed()	Return x Start Speed
+ void setxStartSpeed(double xStartSpeed)	Set x Start Speed
+ double getyStartSpeed()	Return y Start Speed
+ void setyStartSpeed(double yStartSpeed)	Set y Start Speed
+ boolean isMovingThreadStart()	Return true if Moving Thread isn't end
+ void setMovingThreadStart(boolean startThread)	Set Moving Thread Start
+ int getCurrentHp()	Return Current Hp
+ void setCurrentHp(int hp)	Set Current Hp
+ boolean isDead()	Return true if Current Hp == 0
+ int getStartHp()	Return Start Hp
+ void setStartHp(int hp)	Set Start Hp
+ float getHealthRatio()	Return Current Hp / Start Hp
+ int getWidthHpBar()	Return Width of Hp bar
+ void initialHpBar(int h, int w, int x, int y)	Create Hp bar with the given width bar, height bar and position bar
+ void receiveDamage(Object bullet)	If enemy is not immune, calculate current hp after receiving damage and update Hp bar and set enemy immune for 20 ms if bullet can penetrate
+ boolean isImmune()	Return true if enemy immune
+ void setImmune(boolean immune)	Set immune
+ boolean needRespawn()	Return true if enemy need respawn
+ int getBite()	Return bite damage
+ void setBite(int bite)	Set bite damage
+ int getScore()	Return score
+ void setScore(int score)	Set score

2.2.2. Class KingViperPane extends EnemyPane

2.2.2.1. Fields

- Image TEXTURES_FILE	Store texture file
- Map<Integer, Image> *walkingTexturesMap*	Store walking animation
- Map<Integer, Image> *attackTexturesMap*	Store attacking animation
- Map<Integer, Image> *deathTexturesMap*	Store death animation

2.2.2.2. Constructors

+ KingViperPane(int x, int y)	Construct King Viper with the given position and initialize texture number, texture animation, score, Start Hp, Current Hp, Hp Bar, respawn, speed, bite damage, and size of enemy

2.2.2.3. Methods

- int playAnimation(int animationState, int id, Map<Integer, Image> texturesMap, int animationFrame)	Set state to animation state and show animation $(id)^{th}$ frame
+ void animate()	Create Thread for playing animation and start that thread and set respawn to true if animation is end

2.3. package gui.floor

2.3.1. Class FloorPane extends GridPane

2.3.1.1.

- ArrayList<FloorSquare> allCells	Array for collecting FloorSquare and showing texture

2.3.1.2. Constructors

+ FloorPane()	Construct map and initialize texture, size and alignment

2.3.2. Class FloorSquare extends Pane

2.3.2.1. Fields

- Image TEXTURES_FILE	Store texture file
- Map<Integer, Image> texturesMap	Store mapping cropped texture with number key
- int type	Store type of the floor pane square
- int xPositon	x Position of Floor square
- int yPosition	y Position of Floor square

2.3.2.2. Constructors

+ FloorSquare(int x, int y, int type)	Construct FloorSquare with the given position and initialize

2.3.2.3. Methods

Remaining getter/setters for the fields	

2.4. package gui.player

2.4.1. Class PlayerPane extends Pane implements Deadable, Movable, Controllable, Animation

2.4.1.1. Fields

- Image TEXTURES_FILE	Store texture file
- Map<Integer, Image> texturesMap	Store the texture image in map
- double xTranslation	x Position of player
- double yTranslation	y Position of player
- double xSpeed	x Speed of player
- double ySpeed	y Speed of player
- boolean pressedW	True if W key is being pressed
- boolean pressedA	True if A key is being pressed
- boolean pressedS	True if S key is being pressed
- boolean pressedD	True if D key is being pressed
- int currentHp	Current Hp of player
- int startHp	Start Hp of player
- Rectangle statusStartHealthBar	Show damage that player take
- Rectangle statusCurrentHealthBar	Show player's current Hp
- boolean startThread	True if Thread isn't stop
- boolean immune	True if player immune

2.4.1.2. Constructors

+ PlayerPane(int x, int y)	Construct player with the given position and initialize speed, start Hp, current Hp, immune, pressW, pressA, pressS, pressD, Hp bar, and size

2.4.1.3. Methods

+ void animate()	Create Thread for playing the player animation
+ void run()	Create Thread for calculate player position via speed and updating its position and run thread
+ double getxTranslation()	Return x Position
+ void setxTranslation(double xTranslation)	Set x Position
+ double getyTranslation()	Return y Position
+ void setyTranslation(double yTranslation)	Set y Position
+ double getCurrentXPosition()	Return x Position on screen
+ double getCurrentYPosition()	Return y Position on screen
+ double getxSpeed()	Return x Speed
+ void setxSpeed(double speed)	Set x Speed
+ double getySpeed()	Return y Speed
+ void setySpeed(double speed)	Set y Speed
+ boolean isMovingThreadStart()	Return true if thread isn't stop
+ void setMovingThreadStart(boolean startThread)	Set MovingThread

+ int getCurrentHp()	Return current Hp
+ void setCurrentHp(int hp)	Set current Hp
+ boolean isDead()	Return true if current Hp == 0
+ int getStartHp()	Return start Hp
+ void setStartHp(int hp)	Set start Hp
+ float getHealthRatio()	Return current Hp / start Hp
+ int getWidthHpBar()	Return Hp bar
+ void initialHpBar(int h, int w, int x, int y)	Create Hp bar with the given width bar, height bar and position bar
+ void receiveDamage(Object damage)	Set current Hp after receiving damage and update Hp bar
+ boolean isImmune()	Return true if player is immune
+ void setImmune(boolean immune)	Set immune
+ void checkHeal()	If player can kill viper without taking damage, current Hp +1 and update Hp bar
+ void keyPressHandler(KeyEvent key)	- If player press W, set pressW to true - If player press W, set pressA to true and Flip player to left - If player press W, set pressS to true - If player press W, set pressD to true and Flip player to left
+ void keyReleasedHandler(KeyEvent key)	- If player release W, set pressW to false - If player release W, set pressA to false - If player release W, set pressS to false - If player release W, set pressD to false
getter/ setter for pressedW, pressedA, pressedS, pressedD	

2.5. package gui.score

2.5.1. Class ScorePane extends VBox

2.5.1.1. Fields

- Label score	Label for showing score and score multiplier

2.5.1.2. Constructors

+ ScorePane()	Construct ScorePane and initialize label text from score and multiplier, position, text font, text color, size, alignment, and background color

2.5.1.3. Methods

+ void updateScore()	Update text to show current score and score multiplier and change text color to: White if score multiplier == 1 Yellow if score multiplier == 2 Orange if score multiplier == 3 Red if 3<score multiplier<10 Magenta if 10<=score multiplier<20 Light blue if score multiplier == 20
+ void run()	Create Thread for looping updateScore methodand start thread

2.6. package gui.timer

2.6.1. Class GameTimerPane extends Vbox

2.6.1.1. Fields

- Label time	Label for showing time

2.6.1.2. Constructors

+ GameTimerPane(String gameTimer)	Create TimerPane with the given text and initialize position, text font, text color, size, alignment, and background color

2.6.1.3. Methods

+ void updateGameTimer(String gameTimer)	Set text label

2.7. package gui.weapon.bullet

2.7.1. Class Bullet extends Pane implements Movable

2.7.1.1. Fields

- double xTranslation	x Position
- double yTranslation	y Position
- double xSpeed	x Speed
- double ySpeed	y Speed
- int damage	Bullet damage
- boolean isCollision	True when bullet collapse with enemy
- boolean isPenetrate	True when bullet can penetrate enemy
- boolean startThread	True when thread isn't stop

2.7.1.2. Constructors

+ Bullet(double x, double y)	Construct bullet

2.7.1.3. Methods

+ void run()	Create Thread for Calculate bullet position via speed and updating its position and run thread
+ void update(StackPane root)	If bullet out of map, delete bullet from root
+ void calcualteRotation(double xFactor, double yFactor, double xLength, double yLength)	Calculate angle that bullet should rotate and rotate it
+ adjustSpeed(int baseSpeed, double x, double y)	Adjust speed with mouse click direction
+ void setTexture(String imagePath)	Set texture with the given texture
+ double getxTranslation()	Return x Position
+ void setxTranslation(double xTranslation)	Set x Position
+ double getyTranslation()	Return y Position
+ void setyTranslation(double yTranslation)	Set y Position
+ double getCurrentXPosition()	Return x Position on screen
+ double getCurrentYPosition()	Return y Position on screen
+ double getxSpeed()	Return x Speed
+ void setxSpeed(double speed)	Set x Speed
+ double getySpeed()	Return y Speed
+ void setySpeed(double speed)	Set y Speed
+ boolean isMovingThreadStart()	Return true if Moving isn't stop
+ void setMovingThreadStart(boolean startThread)	Set MovingThreadStart
+ boolean outOfMap()	Return true if bullet out of map
getter/ setter for damage, isCollision	

2.7.2. Class FlameThrowerGunBullet extends Bullet

2.7.2.1. Constructors

+ FlameThrowerGunBullet(double x, double y)	Construct Flamethrower bullet with the given position and initialize damage, penetrate, speed, texture, and rotate

2.7.3. Class LaserGunBullet extends Bullet

2.7.3.1. Constructors

+ LaserGunBullet(double x, double y)	Construct Laser gun bullet with the given position and initialize damage, penetrate, speed, texture, and rotate

2.7.4. Class MachineGunBullet extends Bullet

2.7.4.1. Constructors

+ MachineGunBullet(double x, double y)	Construct Machine gun bullet with the given position and initialize damage, penetrate, speed, texture, and rotate

2.7.5. Class PistolGunBullet extends Bullet

2.7.5.1. Constructors

+ PistolGunBullet(double x, double y)	Construct Pistol gun bullet with the given position and initialize damage, penetrate, speed, texture, and rotate

2.7.6. Class RifleGunBullet extends Bullet

2.7.6.1. Constructors

+ RifleGunBullet(double x, double y)	Construct Rifle gun bullet with the given position and initialize damage, penetrate, speed, texture, and rotate

2.7.6.2.

2.7.7. Class ShotGunBullet extends Bullet

2.7.7.1. Constructors

+ ShotGunBullet(double x, double y)	Construct Shotgun bullet with the given position and initialize damage, penetrate, speed, texture, and rotate

2.8. package gui.weapon.inventory

2.8.1. Class WeaponInventory extends GridPane

2.8.1.1. Fields

- ArrayList<String> GUN_TEXTURES	Store the string path of all weapon in the game
- ArrayList<WeaponInventorySquare> weaponInventorySquares	Store all WeaponInventorySquare that generated

2.8.1.2. Constructors

+ WeaponInventory()	Construct WeaponInventory and initialize weaponInventorySquares

2.8.1.3. Methods

+ void update()	Create Thread for update available weapon and equipping weapon and run thread

2.8.2. Class WeaponInventorySquare extends Pane

2.8.2.1. Fields

- String name	Store the weapon name specify to this square

2.8.2.2. Constructors

+ WeaponInventorySquare(int id, String name)	Construct WeaponInventorySquare via id and name

2.8.2.3. Methods

getter/ setter for name	

2.9. package gui.weapon.weapon

2.9.1. Abstract Class Weapon extends Pane implements Movable

2.9.1.1. Fields

- double xTranslation	x Position
- double yTranslation	y Position
- double xSpeed	x Speed
- double ySpeed	y Speed
# xShootingPosition	x Shooting position
# yShootingPosition	y Shooting position
- boolean alreadyShoot	True if gun is already shoot
- int fireRate	Fire rate of the gun
- String shootingFileName	Store file path of texture

2.9.1.2. Constructors

+ Weapon(int x, int y)	Construct weapon with the given position and initialize size and fire rate

2.9.1.3. Methods

+ void run()	Update position on screen
+ void startShoot()	
+ void runShoot(StackPane root)	
+ void endShoot()	Set shooting to false
+ void setTexture(String imagePath)	Set texture of the weapon
+ double getxTranslation()	Return x Position
+ void setxTranslation(double xTranslation)	Set x Position
+ double getyTranslation()	Return y Position
+ void setyTranslation(double yTranslation)	Set y Position
+ double getCurrentXPosition()	Return x Position on screen
+ double getCurrentYPosition()	Return y Position on screen
+ double getxSpeed()	Return x Speed
+ void setxSpeed(double speed)	Set x Speed
+ double getySpeed()	Return y Speed
+ void setySpeed(double speed)	Set y Speed
+ boolean isMovingThreadStart()	Return true if Thread isn't stop
+ void setMovingThreadStart(boolean startThread)	Set MovingThreadStart
+ void setShootPosition(MouseEvent event)	Set shoot position by using the position of mouse
+ void initializeShootingSound(String path)	Initialize sound weapon
+ void playShootingSound()	Play sound weapon
getter/ setter for shooting, alreadyShoot, fireRate	

2.9.2. Class ElderflameGun extends Weapon

2.9.2.1. Constructors

+ ElderflameGun(int x, int y)	Construct Elderflame gun with the given position and initialize fire rate, texture and sound shooting

2.9.2.2. Methods

+ void startShoot()	Set shooting to true
+ void runShoot(StackPane root)	Construct bullet of that weapon while weapon is shooting and add bullet to the root

2.9.3. Class FlameThrowerGun extends Weapon

2.9.3.1. Constructors

+ FlameThrowerGun(int x, int y)	Construct Flamethrower with the given position and initialize fire rate, texture and sound shooting

2.9.3.2. Methods

+ void startShoot()	Set shooting to true
+ void runShoot(StackPane root)	Construct bullet of that weapon while weapon is shooting and add bullet to the root

2.9.4. Class LaserGun extends Weapon

2.9.4.1. Constructors

+ LaserGun(int x, int y)	Construct Laser gun with the given position and initialize fire rate, texture and sound shooting

2.9.4.2. Methods

+ void startShoot()	Set shooting to true
+ void runShoot(StackPane root)	Construct bullet of that weapon while weapon is shooting and add bullet to the root

2.9.5. Class MachineGun extends Weapon

2.9.5.1. Constructors

+ MachineGun(int x, int y)	Construct Machine gun with the given position and initialize fire rate, texture and sound shooting

2.9.5.2. Methods

+ void startShoot()	Set shooting to true
+ void runShoot(StackPane root)	Construct bullet of that weapon while weapon is shooting and add bullet to the root

2.9.6. Class RifleGun extends Weapon

2.9.6.1. Constructors

+ RifleGun(int x, int y)	Construct Rifle gun with the given position and initialize fire rate, texture and sound shooting

2.9.6.2. Methods

+ void startShoot()	Set shooting to true
+ void runShoot(StackPane root)	Construct bullet of that weapon while weapon is shooting and add bullet to the root

2.9.7. Class ShotGun extends Weapon

2.9.7.1. Constructors

+ ShotGun(int x, int y)	Construct Shotgun with the given position and initialize fire rate, texture and sound shooting

2.9.7.2. Methods

+ void startShoot()	Set shooting to true
+ void runShoot(StackPane root)	Construct bullet of that weapon while weapon is shooting and add bullet to the root
- double[] calculateBulletPosition()	Return array of position of all 3 bullets

2.10. package logic

2.10.1. Interface Animation

2.10.1.1. Methods

+ animate()	Play animation

2.10.2. Interface Controllable

2.10.2.1. Methods

+ void keyPressHandler(KeyEvent key)	- If player press W, set pressW to true - If player press W, set pressA to true - If player press W, set pressS to true - If player press W, set pressD to true
+ void keyReleasedHandler(KeyEvent key)	- If player release W, set pressW to false - If player release W, set pressA to false - If player release W, set pressS to false - If player release W, set pressD to false
+ boolean isPressedW()	Return true if W key is being pressed
+ void setPressedW(boolean pressedW)	Set PressedW
+ boolean isPressedA()	Return true if A key is being pressed
+ void setPressedA(boolean pressedA)	Set PressedA
+ boolean isPressedS()	Return true if S key is being pressed
+ void setPressedS(boolean pressedS)	Set PressedS
+ boolean isPressedD()	Return true if D key is being pressed
+ void setPressedD(boolean pressedD)	Set PressedD

2.10.3. Interface Deadable

2.10.3.1. Methods

+ int getCurrentHp()	Return current Hp
+ void setCurrentHp(int hp)	Set current Hp
+ boolean isDead()	Return true if current Hp == 0
+ int getStartHp()	Return start Hp
+ void setStartHp(int hp)	Set start Hp
+ float getHealthRatio()	Return current Hp / start Hp
+ int getWidthHpBar()	Return Width of Hp bar
+ void initialHpBar(int h, int w, int x, int y)	Initialize Hp bar with the give height, width and position
+ void receiveDamage(Object o)	Update current Hp after receiving damage and update Hp bar
+ boolean isImmune()	Return true if immune
+ void setImmune(boolean immune)	Set immune

2.10.4. Interface Movable

2.10.4.1. Methods

+ void run()	Update position
+ double getxTranslation()	Return x Position
+ void setxTranslation(double xTranslation)	Set x Position
+ double getyTranslation()	Return y Position
+ void setyTranslation(double yTranslation)	Set y Position
+ double getCurrentXPosition()	Return x Position on screen
+ double getCurrentYPosition()	Return y Position on screen
+ double getxSpeed()	Return x Speed
+ void setxSpeed(double speed)	Set x Speed
+ double getySpeed()	Return y Speed
+ void setySpeed(double speed)	Set y Speed
+ boolean isMovingThreadStart()	Return true if Thread isn't stop
+ void setMovingThreadStart(boolean startThread)	Set MovingThreadStart

2.10.5. Class GameTimer

2.10.5.1. Fields

- int sec	Variable for storing second
- int min	Variable for storing minute
- boolean timerControl	Set true to start timer

2.10.5.2. Constructors

+ GameTimer()	Construct timer and set minute and second to 0
+ GameTimer(int min, int sec)	Construct timer with the given minute and second

2.10.5.3. Methods

+ timer()	Timer starts
+ toString()	Return time in format "min : sec"
getter/ setter for sec, min, timerControl	

2.10.6. Class GameLogic

2.10.6.1. Fields

- FloorPane floorPane	Store floor
- PlayerPane playerPane	Store player
- ArrayList<EnemyPane> enemyPanes	Store array of every enemy that isn't dead
- KingViperPane kingViper	Store king Viper
- Weapon elderflameGun	Store elderflame gun
- Weapon flameThrowerGun	Store flamethrower
- Weapon laserGun	Store laser gun
- Weapon machineGun	Store machine gun
- Weapon pistolGun	Store pistol
- Weapon rifleGun	Store rifle
- Weapon shotGun	Store shotgun
- Weapon equippedWeapon	Store equipped weapon
- WeaponInventory weaponInventory	Store weapon inventory
- ArrayList<Bullet> bulletList	Store array of bullet
- int score	Store score
- ScorePane scorePane	Store score Pane
- GameTimer gameTimer	Store timer
- GameTimerPane gameTimerPane	Store timer Pane
- boolean startGame	True when game start
- Scene playScene	Store playing scene
- Scene titleScene	Store title scene
- Scene endScene	Store ending scene
- Scene wonScene	Store won scene
- int viperKillStack	Store kill stack
- int multiply	Store score multiplier
- Clip player	Store music player

2.10.6.2. Methods

+ void gameStart(Stage primaryStage, Scene playScene, StackPane root)	Initialize every necessary thing
- void initializeFloor(StackPane root)	Initialize Floor
- void initializePlayer(StackPane root)	Initialize Player Pane
- void initializeEnemy(StackPane root)	Initialize 3 enemies at start
- void initializeViperKing(StackPane root)	Initialize King Viper
- void initializeWeapon(StackPane root)	Initialize all weapons
- void initializeWeaponInventory(StackPane root)	Initialize weapon inventory Pane
- void initializeBullet()	Initialize bullet
- void initializeScorePane(StackPane root)	Initialize Score Pane
- void startUpdateTimeThread(Stage primaryStage, StackPane root)	Initialize timer and timer Pane, and start timer
- void startCheckPlayerGetBiteThread(Stage primaryStage, StackPane root)	- If enemy can bite player, calculate and set current Hp and update Hp bar, reset kill stack, update score multiplier and set player immune for 2 second - If player is dead, stop game, play ending background music and set ending scene
- void startViperSpawnThread(Stage primaryStage, StackPane root)	- Spawn new enemy every 20 seconds, If the time is less than 5 minutes - Initialize Viper King, If time greater than 5 minutes
- void startViperRespawnThread(Stage primaryStage, StackPane root)	- If no enemy is left, end the game. Else, check if enemy is need respawn. If there any viper need respawn, increase player score and multiplier also check player can heal, delete that viper and construct the new one and add to root, and update enemy
- void addControlEventHandler(StackPane root)	Add key pressed and key release event handler for controlling player and switching weapon
- void addShootEventHandler(StackPane root)	Add mouse pressed, mouse dragged and mouse release handler for controlling shoot the weapon
+ FloorPane getFloorPane()	Return FloorPane
+ setFloorPane(FloorPane floorPane)	Set FloorPane
+ void changeWeaponToElderflameGun(StackPane root)	Change equipped weapon to elderflame gun
+ void changeWeaponToFlameThrowerGun(StackPane root)	Change equipped weapon to flamethrower
+ void changeWeaponToLaserGun(StackPane root)	Change equipped weapon to laser gun

+ void changeWeaponToMachineGun(StackPane root)	Change equipped weapon to machine gun
+ void changeWeaponToPistolGun(StackPane root)	Change equipped weapon to pistol
+ void changeWeaponToRifleGun(StackPane root)	Change equipped weapon to rifle
+ void changeWeaponToShotGun(StackPane root)	Change equipped weapon to shotgun
+ boolean overlaps(double point1, double length1, double point2, double length2)	Return true when 2 objects are overlapping
+ boolean isCollision(EnemyPane enemy, Bullet bullet)	Return true if bullet collide enemy
+ boolean canEnemyBite(EnemyPane enemy)	If enemy can bite player, set current Hp and Hp bar after receiving damage and return true
+ void playerGetBite(EnemyPane enemy)	Set current Hp and Hp bar after receiving damage
+ int[] getSpawnPoint()	Return array of random x position and random y position for spawning viper
+ void gameStop(StackPane root)	Stop all thread by set every thread control to false and clear root
+ int getMultiply()	Return score multiplier
+ void setMultiply(int multiply)	Set score multiplier
+ void updateMultiply()	Set score multiplier to the current score multiplier
+ void playBackgroundMusic(String path)	Play background music of the given path
+ void stopBackgroundMusic()	Stop playing background music that is playing now
+ void delay(int ms)	Set thread sleep for the given time
+ boolean setHighScore()	If save file doesn't exist, create new save file and save new high score and return true. Else, read save file and check if score is greater than high score from the save file, over write old high score and return true. Else, return false
+ String getHighScore()	Read save file and return "Highest Score : high score". If save file doesn't exist, return "Highest Score : None"
+ Font font(double size)	Return Font VT323 with the given size
Remaining getter/setters for the fields	