Snake the Explorer

Created by
Nattapong Anansomsin 6431317721
Waranthorn Chansawang 6432154921

2110215 Programming Methodology
Semester 2 Year 2021
Chulalongkorn University

Snake the Explorer

Introduction

Snake is inspired by the classic snake game. But this game is not like an original snake game. Because there will be things that are both challenging and disruptive to players but it also has something to help the player to win this game. All of which will come in the form of items and objects.

Rule

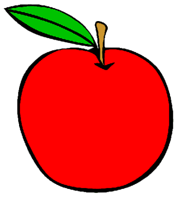
Player controls the snake to move in 4 directions using the W, A, S, and D keys. Snake has a limited amount of "energy" to move, so the player should think carefully about which direction to steer. The more you play through each level, the more difficult it becomes because there are objects which will cause the snake to die and your game is over. In addition to having objects that disrupt you, there are also items that can both help you and disrupt you, so you need to look carefully during play. The objective of this game is to control the snake to collect apples to get the required score in each level. Each level has different score requirements. The player must complete all levels.

Snake

Player controls this character using W, A, S, and D. Snake has only 100 stamina, If the stamina runs out, the player will immediately game over.

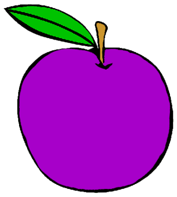
Food

Apple



After collecting, the player will receive 1 point, energy will be reset, and the snake will grow longer.

Bad Apple



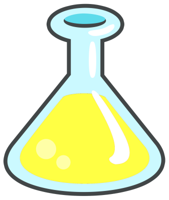
After collecting, the player will receive -1 point but the snake will grow longer like collecting Apple.

Item

Energy

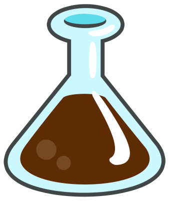
After collecting, the snake will gain 20 more energy for moving.

Speed potion



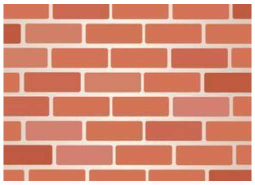
After collecting this potion, the speed of the snake will increase.

Slow potion



After collecting this potion, the speed of the snake will decrease.

Wall



If the snake hits this wall, it will die and the player will immediately game over.

Demon

If the snake hits this monster, it will die and the player will immediately game over.

Peashooter

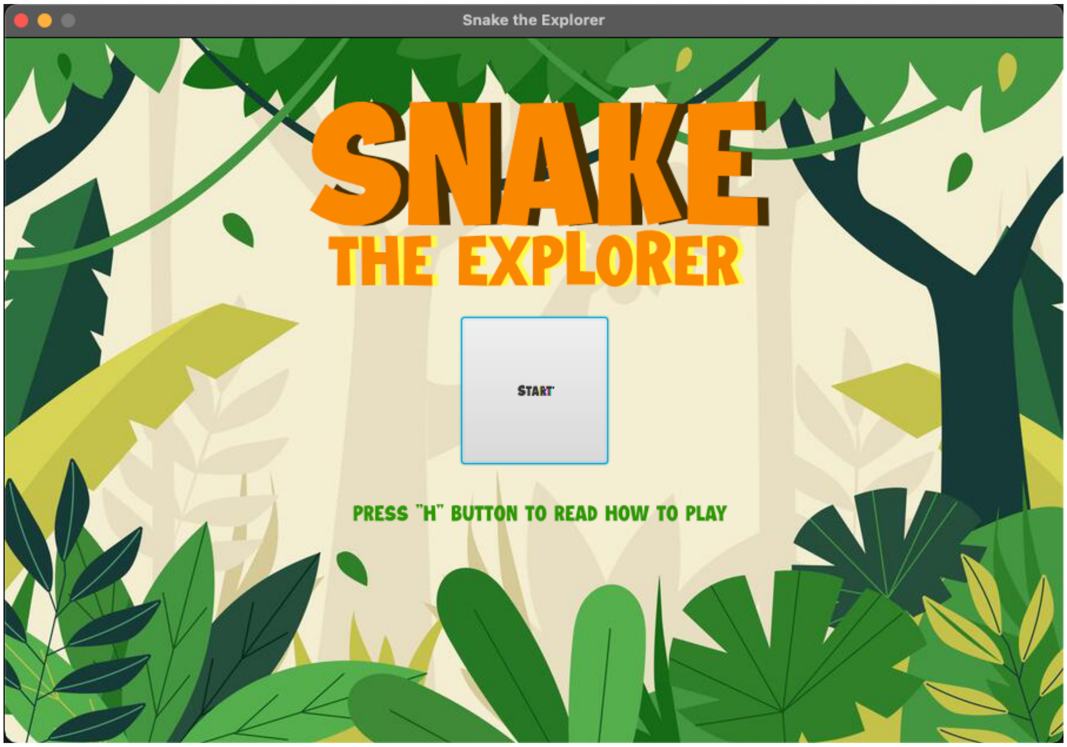
If the snake hits this monster, it will die and the player will immediately game over as same as hitting the Demon but this monster can fire green bullets at the player.

Bullet

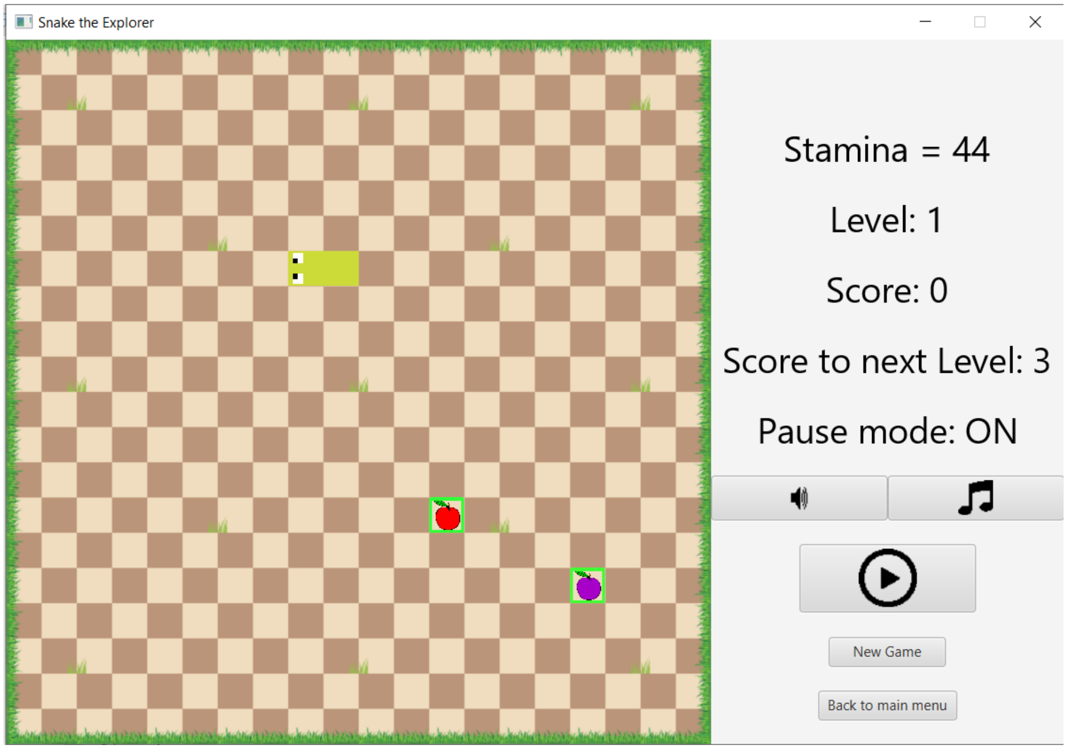
The bullets can move forward, so the player needs to control the snake to avoid these bullets. If the snake hits this bullet, it will die and the player will immediately game over.

Example

Main Menu scene



Level 1



Stamina = 44

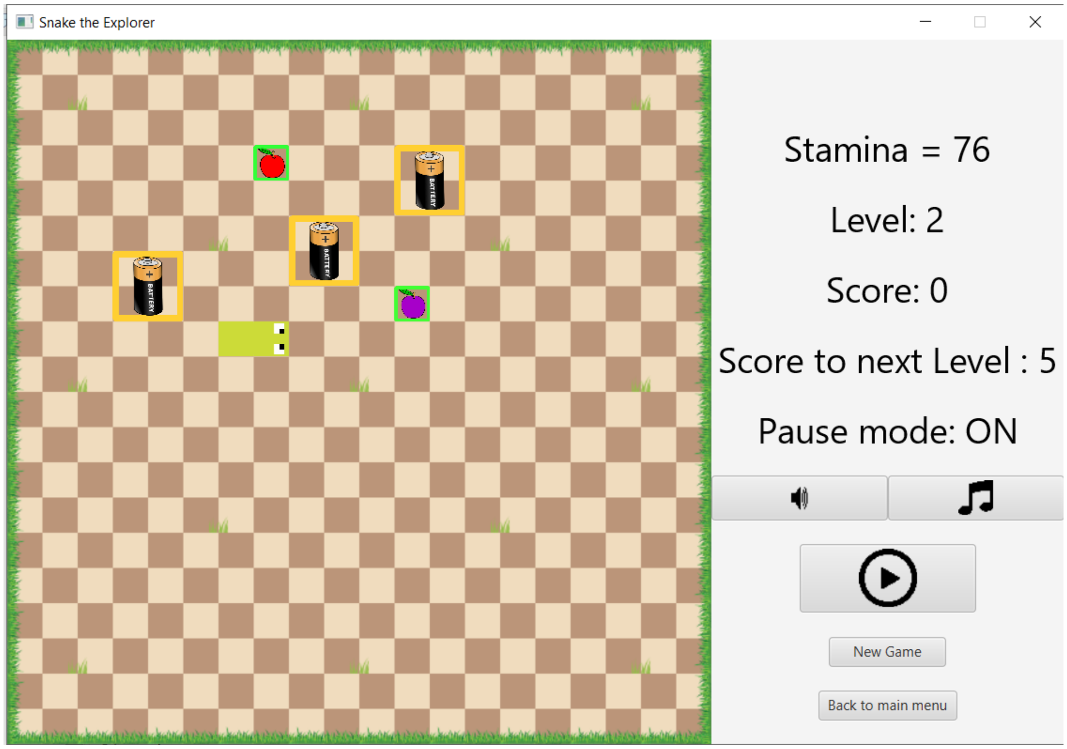
Level: 1

Score: 0

Score to next Level: 3

Pause mode: ON

Level 2



Stamina = 76

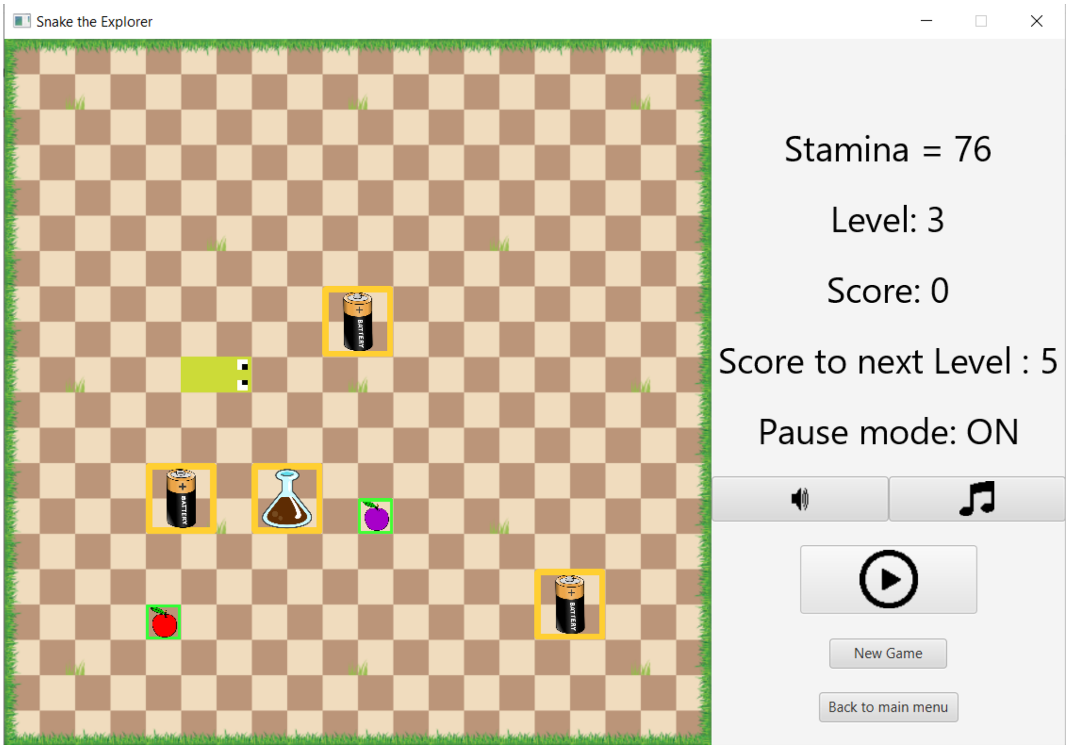
Level: 2

Score: 0

Score to next Level : 5

Pause mode: ON

Level 3



Stamina = 76

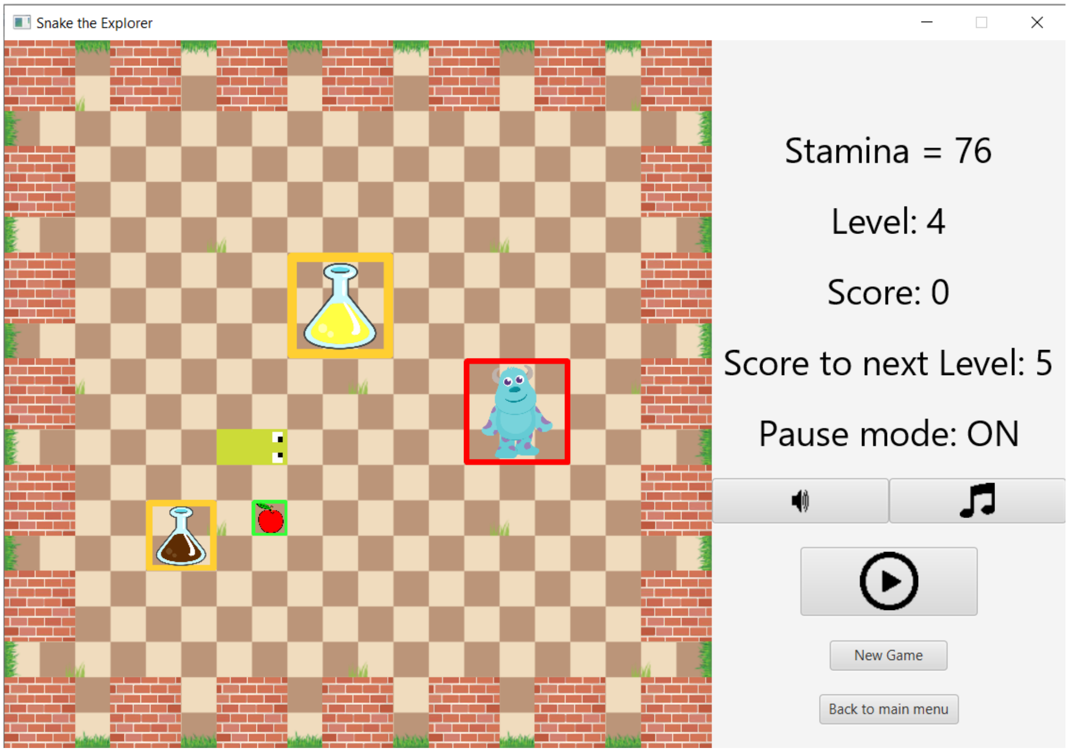
Level: 3

Score: 0

Score to next Level : 5

Pause mode: ON

Level 4



Stamina = 76

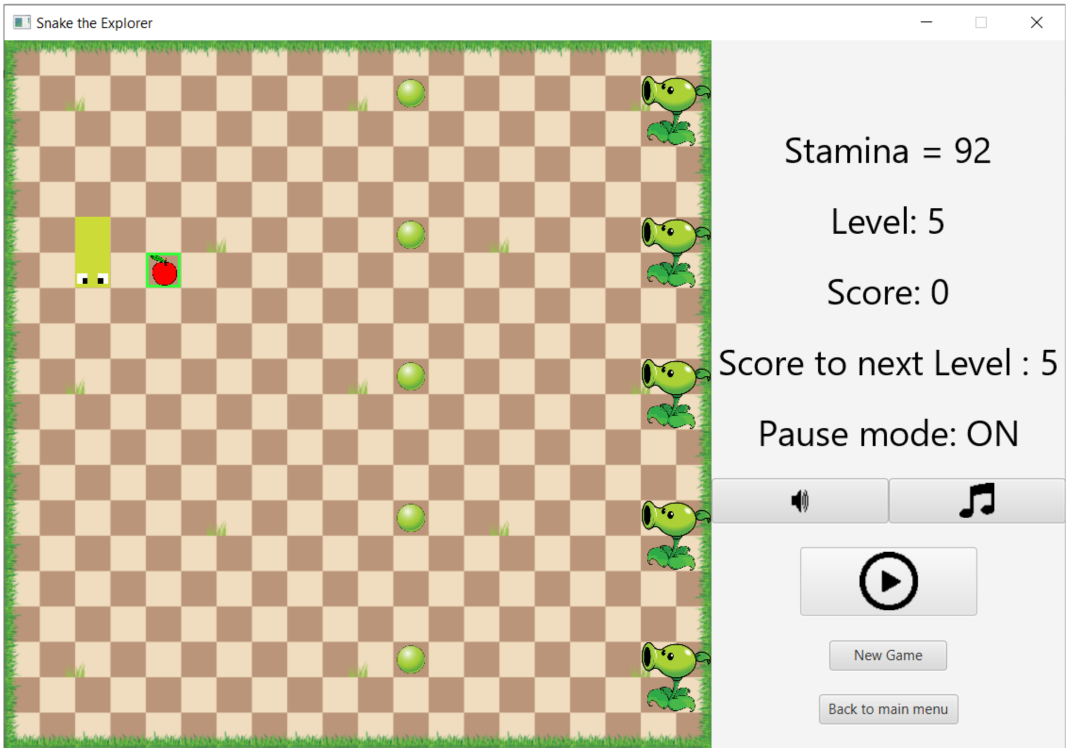
Level: 4

Score: 0

Score to next Level: 5

Pause mode: ON

Level 5



Stamina = 92

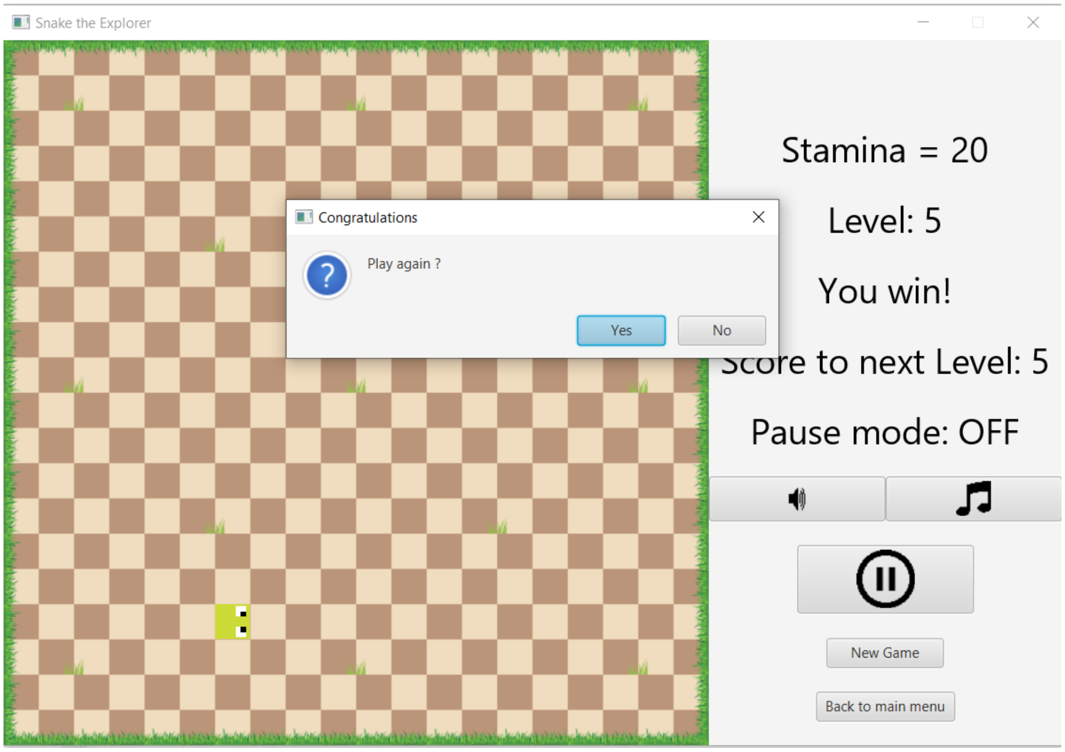
Level: 5

Score: 0

Score to next Level : 5

Pause mode: ON

Game Win



Play again ?

Stamina = 20

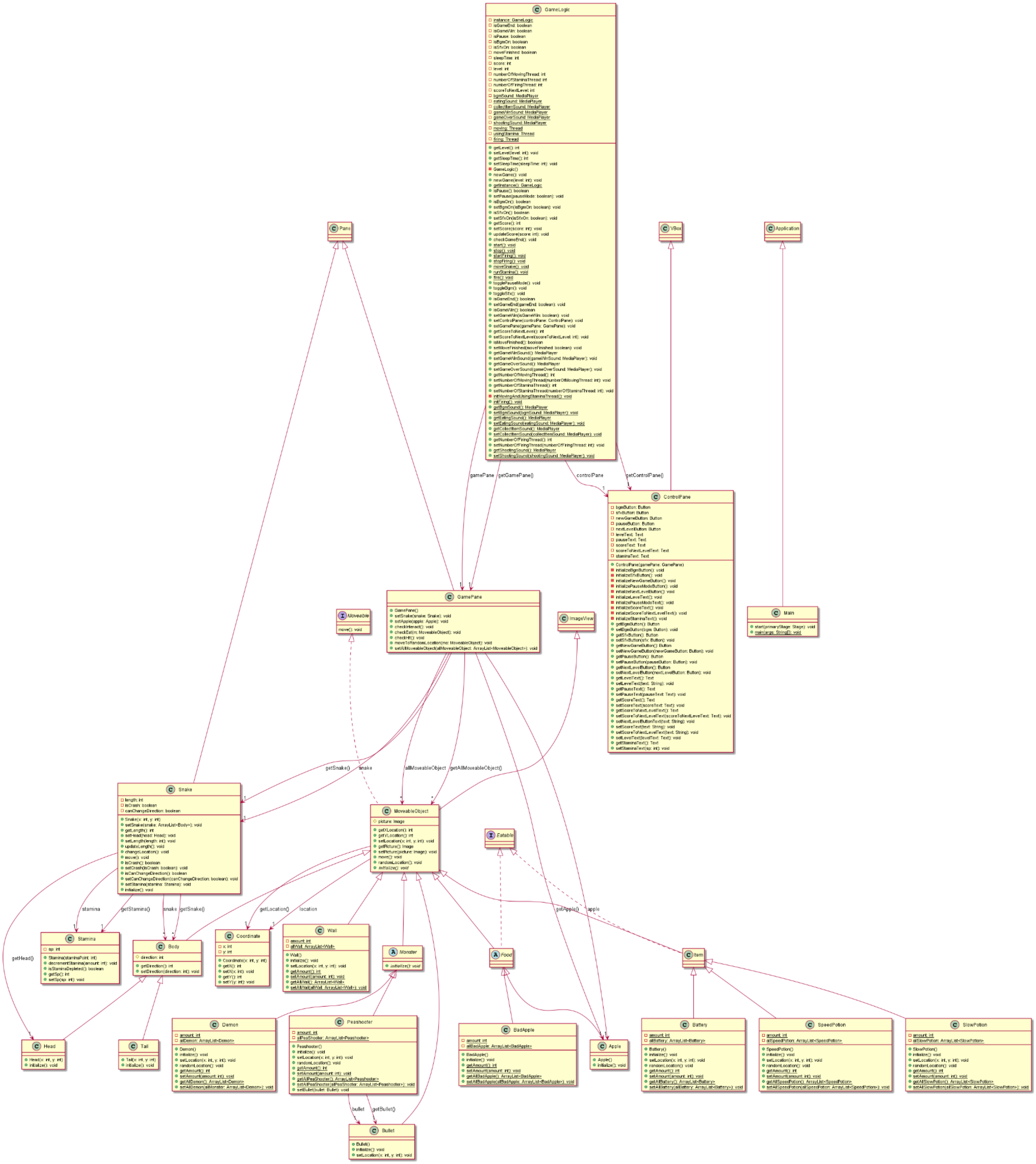
Level: 5

You win!

Score to next Level: 5

Pause mode: OFF

Class diagram


GameLogic
instance: GameLogic
isGameEnd: boolean
isGameWin: boolean
isPause: boolean
isBgmOn: boolean
isSfxOn: boolean
moveFinished: boolean
sleepTime: int
score: int
level: int
numberOfMovingThread: int
numberOfStaminaThread: int
numberOfFiringThread: int
scoreToNextLevel: int
bgmSound: MediaPlayer
eatingSound: MediaPlayer
collectItemSound: MediaPlayer
gameWinSound: MediaPlayer
gameOverSound: MediaPlayer
shootingSound: MediaPlayer
moving: Thread
usingStamina: Thread
firing: Thread
getLevel(): int
setLevel(level: int): void
getSleepTime(): int
setSleepTime(sleepTime: int): void
GameLogic()
newGame(): void
newGame(level: int): void
getInstance(): GameLogic
isPause(): boolean
setPause(pauseMode: boolean): void
isBgmOn(): boolean
setBgmOn(isBgmOn: boolean): void
isSfxOn(): boolean
setSfxOn(isSfxOn: boolean): void
getScore(): int
setScore(score: int): void
updateScore(score: int): void
checkGameEnd(): void
start(): void
stop(): void
startFiring(): void
stopFiring(): void
moveSnake(): void
runStamina(): void
fire(): void
togglePauseMode(): void
toggleBgm(): void
toggleSfx(): void
isGameEnd(): boolean
setGameEnd(gameEnd: boolean): void
isGameWin(): boolean
setGameWin(isGameWin: boolean): void
setControlPane(controlPane: ControlPane): void
setGamePane(gamePane: GamePane): void
getScoreToNextLevel(): int
setScoreToNextLevel(scoreToNextLevel: int): void
isMoveFinished(): boolean
setMoveFinished(moveFinished: boolean): void
getGameWinSound(): MediaPlayer
setGameWinSound(gameWinSound: MediaPlayer): void
getGameOverSound(): MediaPlayer
setGameOverSound(gameOverSound: MediaPlayer): void
getNumberOfMovingThread(): int
setNumberOfMovingThread(numberOfMovingThread: int): void
getNumberOfStaminaThread(): int
setNumberOfStaminaThread(numberOfStaminaThread: int): void
initMovingAndUsingStaminaThread(): void
initFiring(): void
getBgmSound(): MediaPlayer
setBgmSound(bgmSound: MediaPlayer): void
getEatingSound(): MediaPlayer
setEatingSound(eatingSound: MediaPlayer): void
getCollectItemSound(): MediaPlayer
setCollectItemSound(collectItemSound: MediaPlayer): void
getNumberOfFiringThread(): int
setNumberOfFiringThread(numberOfFiringThread: int): void
getShootingSound(): MediaPlayer
setShootingSound(shootingSound: MediaPlayer): void
Pane
VBox
Application
gamePane
getGamePane()
controlPane
getControlPane()
1
1
ControlPane
bgmButton: Button
sfxButton: Button
newGameButton: Button
pauseButton: Button
nextLevelButton: Button
levelText: Text
pauseText: Text
scoreText: Text
scoreToNextLevelText: Text
staminaText: Text
ControlPane(gamePane: GamePane)
initializeBgmButton(): void
initializeSfxButton(): void
initializeNewGameButton(): void
initializePauseModeButton(): void
initializeNextLevelButton(): void
initializeLevelText(): void
initializePauseModeText(): void
initializeScoreText(): void
initializeScoreToNextLevelText(): void
initializeStaminaText(): void
getBgmButton(): Button
setBgmButton(bgm: Button): void
getSfxButton(): Button
setSfxButton(sfx: Button): void
getNewGameButton(): Button
setNewGameButton(newGameButton: Button): void
getPauseButton(): Button
setPauseButton(pauseButton: Button): void
getNextLevelButton(): Button
setNextLevelButton(nextLevelButton: Button): void
getLevelText(): Text
setLevelText(text: String): void
getPauseText(): Text
setPauseText(pauseText: Text): void
getScoreText(): Text
setScoreText(scoreText: Text): void
getScoreToNextLevelText(): Text
setScoreToNextLevelText(scoreToNextLevelText: Text): void
setNextLevelButtonText(text: String): void
setScoreText(text: String): void
setScoreToNextLevelText(text: String): void
setLevelText(levelText: Text): void
getStaminaText(): Text
setStaminaText(sp: int): void
Main
start(primaryStage: Stage): void
main(args: String[]): void
1
1
GamePane
GamePane()
setSnake(snake: Snake): void
setApple(apple: Apple): void
checkInteract(): void
checkEat(m: MoveableObject): void
checkHit(): void
moveToRandomLocation(mo: MoveableObject): void
setAllMoveableObject(allMoveableObject: ArrayList<MoveableObject>): void
Moveable
move(): void
ImageView
getSnake()
snake
allMoveableObject
getAllMoveableObject()
1
1
*
*
Snake
length: int
isCrash: boolean
canChangeDirection: boolean
Snake(x: int, y: int)
setSnake(snake: ArrayList<Body>): void
getLength(): int
setHead(head: Head): void
setLength(length: int): void
updateLength(): void
changeLocation(): void
move(): void
isCrash(): boolean
setCrash(isCrash: boolean): void
isCanChangeDirection(): boolean
setCanChangeDirection(canChangeDirection: boolean): void
setStamina(stamina: Stamina): void
initialize(): void
MoveableObject
picture: Image
getXLocation(): int
getYLocation(): int
setLocation(x: int, y: int): void
getPicture(): Image
setPicture(picture: Image): void
move(): void
randomLocation(): void
initialize(): void
Eatable
stamina
getStamina()
snake
getSnake()
getLocation()
location
getApple()
apple
getHead()
1
1
*
*
1
1
1
Stamina
sp: int
Stamina(staminaPoint: int)
decrementStamina(amount: int): void
isStaminaDepleted(): boolean
getSp(): int
setSp(sp: int): void
Body
direction: int
getDirection(): int
setDirection(direction: int): void
Coordinate
x: int
y: int
Coordinate(x: int, y: int)
getX(): int
setX(x: int): void
getY(): int
setY(y: int): void
Wall
amount: int
allWall: ArrayList<Wall>
Wall()
initialize(): void
setLocation(x: int, y: int): void
getAmount(): int
setAmount(amount: int): void
getAllWall(): ArrayList<Wall>
setAllWall(allWall: ArrayList<Wall>): void
Monster
initialize(): void
Food
Item
Head
Head(x: int, y: int)
initialize(): void
Tail
Tail(x: int, y: int)
initialize(): void
Demon
amount: int
allDemon: ArrayList<Demon>
Demon()
initialize(): void
setLocation(x: int, y: int): void
randomLocation(): void
getAmount(): int
setAmount(amount: int): void
getAllDemon(): ArrayList<Demon>
setAllDemon(allMonster: ArrayList<Demon>): void
Peashooter
amount: int
allPeaShooter: ArrayList<Peashooter>
Peashooter()
initialize(): void
setLocation(x: int, y: int): void
randomLocation(): void
getAmount(): int
setAmount(amount: int): void
getAllPeaShooter(): ArrayList<Peashooter>
setAllPeaShooter(allPeaShooter: ArrayList<Peashooter>): void
setBullet(bullet: Bullet): void
bullet
getBullet()
1
1
Bullet
Bullet()
initialize(): void
setLocation(x: int, y: int): void
BadApple
amount: int
allBadApple: ArrayList<BadApple>
BadApple()
initialize(): void
getAmount(): int
setAmount(amount: int): void
getAllBadApple(): ArrayList<BadApple>
setAllBadApple(allBadApple: ArrayList<BadApple>): void
1
1
Apple
Apple()
initialize(): void
Battery
amount: int
allBattery: ArrayList<Battery>
Battery()
initialize(): void
setLocation(x: int, y: int): void
randomLocation(): void
getAmount(): int
setAmount(amount: int): void
getAllBattery(): ArrayList<Battery>
setAllBattery(allBattery: ArrayList<Battery>): void
SpeedPotion
amount: int
allSpeedPotion: ArrayList<SpeedPotion>
SpeedPotion()
initialize(): void
setLocation(x: int, y: int): void
randomLocation(): void
getAmount(): int
setAmount(amount: int): void
getAllSpeedPotion(): ArrayList<SpeedPotion>
setAllSpeedPotion(allSpeedPotion: ArrayList<SpeedPotion>): void
SlowPotion
amount: int
allSlowPotion: ArrayList<SlowPotion>
SlowPotion()
initialize(): void
setLocation(x: int, y: int): void
randomLocation(): void
getAmount(): int
setAmount(amount: int): void
getAllSlowPotion(): ArrayList<SlowPotion>
setAllSlowPotion(allSlowPotion: ArrayList<SlowPotion>): void

1. Package main

1.1 class Main extends Application

1.1.1 Field

Name	Description
+void start(Stage primaryStage)	Launch main menu. Show the primaryStage
+void main(String[] args)	The entry point of application.

2. Package base

2.1 class Coordinate

This class represents the location of objects in the game.

2.1.1 Field

Name	Description
-int x	Represents location in X-axis.
-int y	Represents location in Y-axis.

2.1.2 Constructor

Name	Description
+Coordinate(int x,int y)	Set x,y location of Coordinate.

2.1.3 Method

Name	Description
+GETTER & SETTER for each field	

2.2 abstract class MoveableObject extends ImageView implements Moveable

This class represents the base of all MoveableObject in the game.

2.2.1 Field

Name	Description
#Coordinate location	Coordinates of the object.
#Image picture	Picture of the object.

2.2.1 Method

Name	Description
+void move()	Move the object to the corresponding location.

+void randomLocation()	Random location of the object.
+int getXLocation()	Get X location.
+int getYLocation()	Get Y location.
+abstract void initialize()	Initialize the object.
+void setLocation(int x, int y)	Set Location of the object.
+GETTER & SETTER for each field	

2.3 class Stamina

This class represents the stamina of snake.

2.3.1 Field

Name	Description
-int sp	Stamina points of the snake.

2.3.2 Constructor

Name	Description
+Stamina(int staminaPoint)	Set sp to staminaPoint.

2.3.3 Method

Name	Description
+void decrementStamina(int amount)	Decrease the stamina points by the specified amount.
+boolean isStaminaDepleted()	Check if sp <= 0.
+GETTER & SETTER for each field	

3. Package gui

3.1 class GamePane extends Pane

This class represents a game pane.

3.1.1 Field

Name	Description
-Snake snake	Represents snake.
-Apple apple	Represents apple.

-ArrayList<MoveableObject> allMoveableObject	Store all moveable objects in the game Pane.

3.1.2 Constructor

Name	Description
+GamePane()	Initialize all fields and objects in game Pane.

3.1.3 Method

Name	Description
+void checkInteract(int x,int y)	Check the interaction between the snake's head and other objects at coordinates (x,y).
+void checkEat()	Check what is eaten.
+void checkHit()	Check if the snake is hit by the bullet.
+void moveToRandomLocation(Moveable m)	Move object m to a random location.
+GETTER & SETTER for each field	

3.2 class ControlPane extends VBox

This class represents the control pane.

3.2.1 Field

Name	Description
-Button bgmButton	Button for turn on, turn off bgm.
-Button sfxButton	Button for turn on, turn off sfx.
-Button newGameButton	Button for new game.
-Button pauseButton	Button for pause game.
-Button nextLevelButton	Button for go to the next level.
-Text levelText	The text for displaying current level.
-Text pauseText	The text for displaying game is pause or not.
-Text scoreText	The text for displaying current score.
-Text scoreToNextLevelText	The text for displaying score need to go to next level.
-Text staminaText	The text for displaying current stamina.

3.2.2 Constructor

Name	Description

+ControlPane()	Initialize all fields.

3.2.3 Method

Name	Description
-void initializeBgmButton()	Initializes button for toggle background music.
-void initializeSfxButton()	Initializes button for toggle sound effect.
-void initializeNewGameButton()	Initializes button for new game.
-void initializePauseModeButton()	Initializes button for pause game.
-void initializeNextLevelButton()	Initializes button for go to next level.
-void initializeLevelText()	Initializes text for display current level.
-void initializePauseModeText()	Initializes text for display pause mode.
-void initializeScoreText()	Initializes text for display current score.
-void initializeScoreToNextLevelText()	Initializes text for display score to next level.
-void initializeStaminaText()	Initializes text for display current stamina.
+GETTER & SETTER for each field	

4. Package interfaces

4.1 interface Eatable

This interface is a marker interface for objects that can be eaten.

4.2 interface Moveable

This interface is an interface for objects that can be moved.

4.2.1 Method

Name	Description
+void move()	Move the object.

5. Package logic

5.1 class GameLogic

This class represents game logic.

5.1.1 Field

Name	Description
-GameLogic instance	Instance that represents GameLogic class.
-boolean isGameEnd	Determine whether the game is over or not.
-boolean isGameWin	Determine whether the player wins the game

	or not.
-boolean isPause	Determine whether the game is paused or not.
-boolean isBgmOn	Determine whether the bgm is on or not.
-boolean isSfxOn	Determine whether the sfx is on or not.
-boolean moveFinished	Determine whether the snake move is finished or not.
-int sleepTime	Sleep time of moving thread.
-int score	Current score.
-int level	Current level.
-int numberOfMovingThread	Number of moving threads.
-int numberOfStaminaThread	Number of usingStamina thread.
-int numberOfFiringThread	Number of firing threads.
-int scoreToNextLevel	Score to next level.
-GamePane gamePane	Game pane.
-ControlPane controlPane	Control Pane.
-static MediaPlayer bgmSound	Background music.
-static MediaPlayer eatingSound	Sound plays when eating food.
-static MediaPlayer collectItemSound	Sound plays when collecting items.
-static MediaPlayer gameWinSound	Sound plays when winning the game.
-static MediaPlayer gameOverSound	Sound plays when gameover.
-static MediaPlayer shootingSound	Sound plays when a pea shooter shoots the bullet.
-static Thread moving	Thread for moving snake.
-static Thread usingStamina	Thread for reducing the stamina of snakes.
-static Thread firing	Thread that makes the pea shooter shoot the bullet.

5.1.2 Constructor

Name	**Description**

+GameLogic()	Call the newGame method.

5.1.3 Method

Name	Description
+void newGame()	Start a new game.
+void newGame(int level)	Start a new game at the specific level.
+static GameLogic getInstance()	Getter of instance.
+void updateScore(int score)	Update score, score text and check if the game is end by using checkGameEnd() method.
+void checkGameEnd()	Check if the game is end.
+static void start()	Initialize moving thread and usingStaminaThread and start them.
+static void stop()	Interrupt moving thread and usingStamina thread.
+static void startFiring()	Initialize firing thread and start it.
+static void stopFiring()	Interrupt firing thread
+static void moveSnake()	Move the snake.
+static void runStamina()	Decrease the stamina.
+static void fire()	Make the pea shooter shoot the bullet.
+void togglePauseMode()	Toggle pause mode.
+void toggleBgm()	Toggle bgm.
+void toggleSfx()	Toggle sfx.
-static void initMovingAndUsingStaminaThread()	Initialize moving thread and usingStamina thread.
-static void initFiring()	Initialize firing thread.
+GETTER & SETTER for each field	

6. Package food

6.1 abstract class Food extends MoveableObject implement Eatable;

This class is a base class for all food in the game.

6.2 class Apple extends Food

This class represents an apple.

6.2.1 Constructor

Name	Description
+Apple()	Initializes all fields.

6.2.2 Method

Name	Description
+void initialize()	Set visible to true.

6.3 class BadApple extends Food

This class represents a bad apple.

6.3.1 Fields

Name	Description
-static int amount	Amount of bad apples that are visible.
-static ArrayList<BadApple> allBadApple	Store all bad apples that have been created.

6.2.1 Constructor

Name	Description
+BadApple()	Initializes all fields.

6.2.2 Method

Name	Description
+void initialize()	Set visible to true and increase the amount by 1.
+GETTER & SETTER for each field	

7. Package item

7.1 abstract class Item extends MoveableObject implement Eatable;

This class is a base class for all items in the game.

7.2 class Battery extends Item

This class represents a battery that will increase snake stamina when it is eaten.

7.2.1 Fields

Name	Description
-static int amount	Amount of batteries that are visible.
-static ArrayList<Battery> allBattery	Store all batteries that have been created.

7.2.1 Constructor

Name	Description
+Battery()	Initializes all fields.

7.2.3 Method

Name	Description
+void initialize()	Set visible to true and increase the amount by 1.
+void setLocation(int x, int y)	Set location of the object.
+void randomLocation()	Random location of the object.
+GETTER & SETTER for each field	

7.3 class SlowPotion extends Item

This class represents a slow potion that will slow down a snake when it is eaten.

7.3.1 Fields

Name	Description
-static int amount	Amount of slow potion that is visible.
-static ArrayList<SlowPotion> allSlowPotion	Store all slow potions that have been created.

7.3.2 Constructor

Name	Description
+SlowPotion()	Initializes all fields.

7.3.3 Method

Name	Description
+void initialize()	Set visible to true and increase the amount by 1.
+void setLocation(int x, int y)	Set location of the object.
+void randomLocation()	Random location of the object.
+GETTER & SETTER for each field	

7.4 class SpeedPotion extends Item

This class represents a speed potion that will speed up a snake when it is eaten.

7.4.1 Fields

Name	Description
-static int amount	Amount of speed potion that is visible.
-static ArrayList<SpeedPotion> allSpeedPotion	Store all speed potions that have been created.

7.4.2 Constructor

Name	Description
+SpeedPotion()	Initializes all fields.

7.4.3 Method

Name	Description
+void initialize()	Set visible to true and increase the amount by 1.
+void setLocation(int x, int y)	Set location of the object.
+void randomLocation()	Random location of the object.
+GETTER & SETTER for each field	

8. Package Monster

8.1 class Monster extends Item

This class is a base class for all monsters in the game.

8.2 class Demon extends Monster

This class represents demon in the game.

8.2.1 Fields

Name	Description
-static int amount	Amount of demons that is visible.
-static ArrayList<Demon> allDemon	Store all demons that have been created.

8.2.2 Constructor

Name	Description
+Demon()	Initializes all fields.

8.2.3 Method

Name	Description
+void initialize()	Set visible to true and increase the amount by 1.

+void setLocation(int x, int y)	Set location of the object.
+void randomLocation()	Random location of the object.
+GETTER & SETTER for each field	

8.3 class PeaShooter extends Monster

This class represents pea shooters in the game.

8.2.1 Fields

Name	**Description**
-static int amount	Amount of pea shooters that is visible.
-static ArrayList<PeaShooter> allPeaShooter	Store all pea shooters that have been created.
-Bullet bullet	Bullet of the pea shooter.

8.2.2 Constructor

Name	**Description**
+PeaShooter()	Initializes all fields.

8.2.3 Method

Name	**Description**
+void initialize()	Set visible to true and increase the amount by 1.
+void setLocation(int x, int y)	Set location of the object.
+void randomLocation()	Random location of the object.
+GETTER & SETTER for each field	

9. package snake

9.1 abstract class Body extends MoveableObject

This is a base class for the head and tail of the snake.

9.1.1 Fields

Name	**Description**
#int direction	Direction of moving. 0 is left 1 is up 2 is right 3 is down

9.1.2 Method

Name	Description
+GETTER & SETTER for each field	

9.2 class Head extends Body

This class represents the head of the snake.

9.1.1 Constructor

Name	Description
+Head(int x,int y)	Initialize the head at the specific location.

9.1.2 Method

Name	Description
+initialize()	Initialize the head.

9.3 class Tail extends Body

This class represents the head of the snake.

9.3.1 Constructor

Name	Description
+Head(int x,int y)	Initialize the head at the specific location.

9.3.2 Method

Name	Description
+initialize()	Set visible to true.

9.4 class Snake extends Pane

This class represents the snake.

9.4.1 Field

Name	Description
-ArrayList<Body> snake	Store every part of the snake. For example: snake.get(0) is head snake.get(1) is tail 1 snake.get(2) is tail 2
-int length	Represent the length of the snake that is visible.
-boolean isCrash;	Determine whether the snake crashed or not.
-boolean canChangeDirection	Determine whether the snake can change direction or not.
-Stamina stamina	Represent the stamina of the snake

9.4.2 Constructor

Name	Description
+Snake(int x, int y)	Initialize the snake by initializing head and tail at the specified location.

9.4.3 Method

Name	Description
+void initialize()	Initialize the snake.
+void updateLength()	Update the length of the snake that is visible.
+void changeLocation()	Change location of head corresponding to head direction then change the location of each part of the snake to the location of the previous part. For example: Change location of tail1 to location of head. Change location of tail2 to location of tail1
+void move()	Move every part of the snake corresponding to each part's location.

10. Package etc

10.1 Class Bullet extends MoveableObject

This class represents the bullet of the pea shooter.

10.1.2 Constructor

Name	Description
+Bullet()	Initialize all fields.

10.1.3 Method

Name	Description
+void initialize()	Set visible to true.
+void setLocation(int x, int y)	Set Location of the object.

10.2 class Wall extends MoveableObject

This class represents a wall in the game.

10.2.1 Fields

Name	Description
-static int amount	Amount of walls that is visible.
-static ArrayList<Wall > allWall	Store all walls that have been created.

10.2.2 Constructor

Name	Description
+Wall()	Initializes all fields.

10.2.3 Method

Name	Description
+void initialize()	Set visible to true and increase the amount by 1.
+void setLocation(int x, int y)	Set location of the object.
+GETTER & SETTER for each field	