Magical World

Documentation

Presented by

lnwza

Members

Wanatha Chaturarat 6631345421

Pitchaya Arkatvipat 6631338021

Magical World

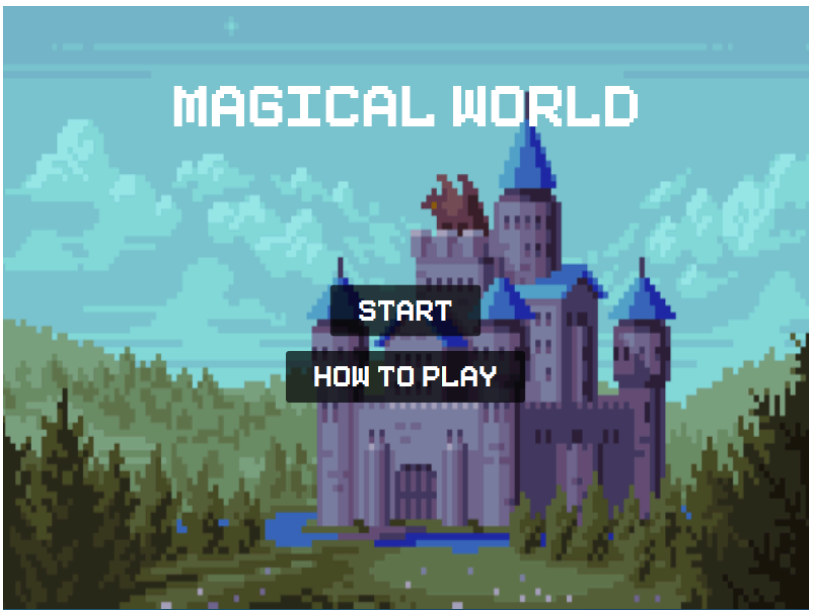
Introduction

This game is an 8-bit rpg game. It is about a witch fighting with monsters in a magical world. The objective is to conquer all monsters in the game.

Game

1.Main menu

This is the main menu of this game. There is a start button to start the game and a how to play button which describe how to play and control the character.



2.How to play

- Press W, A, S, D on the keyboard to control the character.
- Press the spacebar to shoot magic.

3.Items

- heal potion

  -heal player's hp by 5

  -player's hp can't exceeds 20

- mana potion

  -increase player's mana by 5

  -player's mana can't exceeds 20

- power potion

  -increase player damage by 1

  -player's damage can't exceeds 5

- broom

increase player’s speed

- shield

-protected player from being attacked for 10 second

-It cannot be used in item collecting stage

- wand

If the player doesn't have a wand, the player can’t attack.

- key

If the player doesn't have a key. Player cannot go to the next stage.

4.Item collecting stage

This is the item collecting stage. Players can collect useful items before going out and fighting with monsters.

- Players can view inventory and use some items by clicking on the chest icon on the right corner. Some items can’t be used in some stages.
- Player can’t attack in this scene.
- Players must have the key in the inventory slot and walk through the door to go to the next stage.

These are inventory slots.

5.Monster Stage

This is the monster stage.Player must conquer all monsters in this stage before going to the next stage.

- HP and Mana
    - HP and Mana stat is on the left corner
    - Every player’s attacking uses mana.If mana is equal to 0, the player can’t attack.
    - HP and Mana increases every 5 seconds.
    - If a player's HP is equal to 0, the player dies and the game is over.
- Monsters

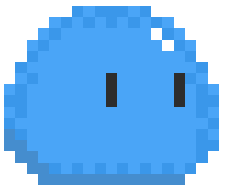

bat golem

    - There are two types of monster in this stage, bat and golem. Each type has different attributes.
    - The monsters are regenerated every 5 seconds randomly.It generated more 5 times before going to the next stage.
    - Monsters drop random items (HP potion, mana potion, power potion and shield) after they are killed.
    - Player’s inventory can’t have more than 9 items.If the sum of player’s item and item in the map is equal to 9, monsters don’t drop more items.

6.Boss Stage

This is the last stage. Player must conquer the boss, SLIME, to win the game.

- Slime



It is the boss to conquer.

- Fire bomb

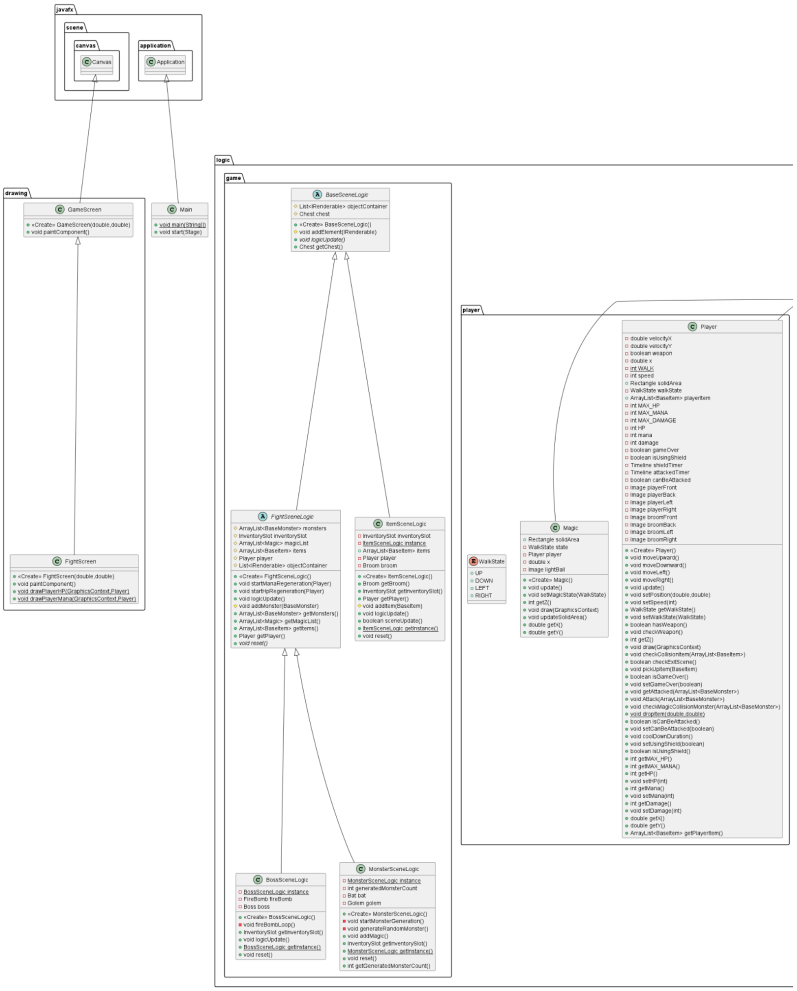
  - It is generated randomly in the map and is generated every 2 seconds.
  - Player can be damage by this fire bomb by being in its area

This page pops up when a player wins the game.

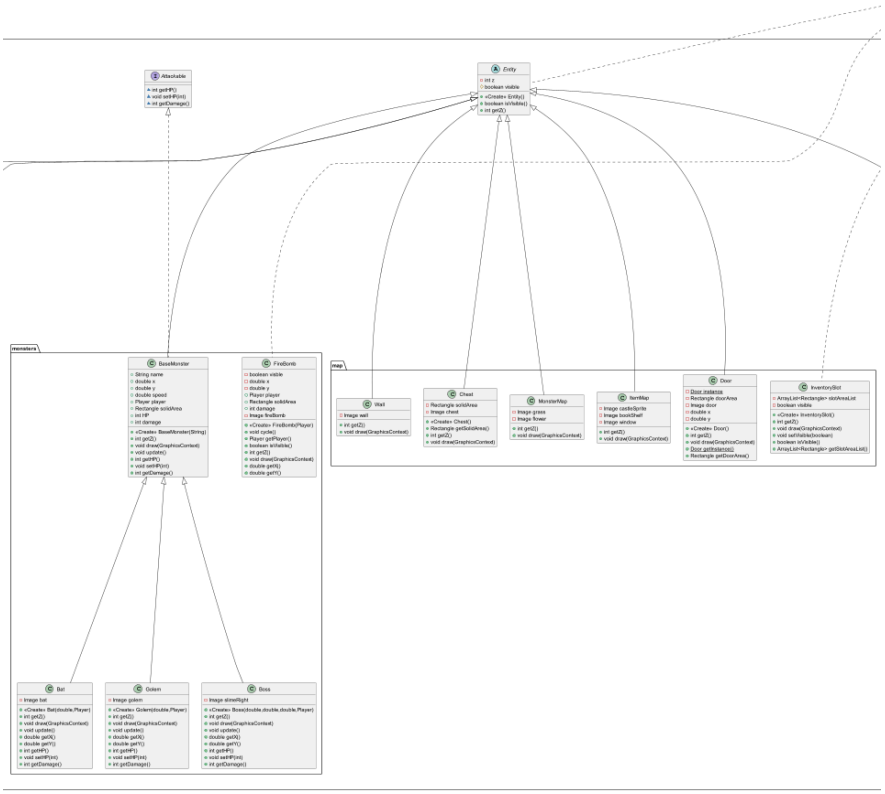
This page pops up when a player loses the game.

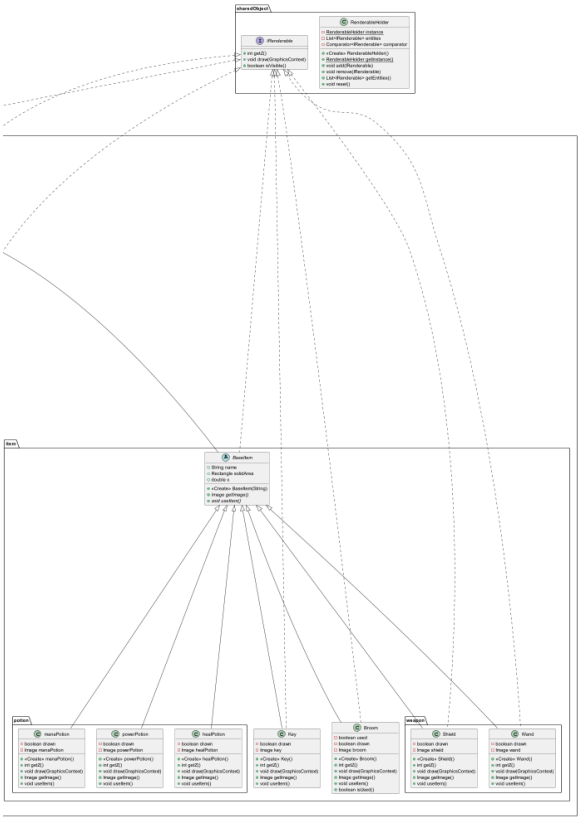
Class Diagram

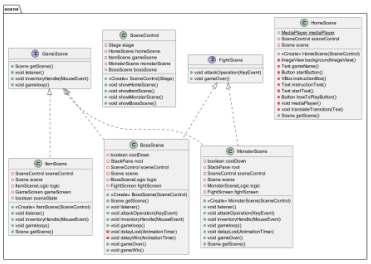
github : https://github.com/2110215-ProgMeth/project-cp-2023-2-lnwza




utils
Config
- double screenWidth
- double screenHeight
- double playerWidth
- double playerHeight
- double playerWidthWithRoom
Input
- ArrayList<KeyCode> keyPressed
+ boolean getKeyPressed(KeyCode)
+ void setKeyPressed(KeyCode, boolean)
+ ArrayList<KeyCode> getKeyPressedList()


Attackable
+ int getHP()
+ void setHP(int)
+ int getDamage()
Entity
- int x
- boolean visible
+ <Create> Entity()
+ boolean isVisible()
+ int getX()
monsters
BossMonster
- String name
- double x
- double y
- double speed
- Player player
- Rectangle soldArea
- int HP
- int damage
+ <Create> BossMonster(String)
+ int getX()
+ void draw(GraphicsContext)
+ void update()
+ int getHP()
+ void setHP(int)
+ int getDamage()
FireBomb
- boolean visible
- double x
- double y
- Player player
- Rectangle soldArea
- int damage
- Image fireBomb
+ <Create> FireBomb(Player)
+ void cycle()
+ Player getPlayer()
+ boolean isVisible()
+ int getX()
+ void draw(GraphicsContext)
+ double getX()
+ double getY()
map
Wall
- Image wall
+ int getZ()
+ void draw(GraphicsContext)
Chest
- Rectangle soldArea
- Image chest
+ <Create> Chest()
+ Rectangle getSoldArea()
+ int getZ()
+ void draw(GraphicsContext)
MonsterMap
- Image grass
- Image flower
+ int getZ()
+ void draw(GraphicsContext)
ItemMap
- Image castleSprite
- Image boatShelf
- Image window
+ int getZ()
+ void draw(GraphicsContext)
Door
- Door instance
- Rectangle doorArea
- Image door
- double x
- double y
+ <Create> Door()
+ int getZ()
+ void draw(GraphicsContext)
- Door getInstance()
+ Rectangle getDoorArea()
InventorySlot
- ArrayList<Rectangle> slotAreaList
- boolean visible
+ <Create> InventorySlot()
+ int getZ()
+ void draw(GraphicsContext)
+ void setVisible(boolean)
+ boolean isVisible()
+ ArrayList<Rectangle> getSlotAreaList()
Bat
- Image bat
+ <Create> Bat(double,Player)
+ int getZ()
+ void draw(GraphicsContext)
+ void update()
+ double getX()
+ double getY()
+ int getHP()
+ void setHP(int)
+ int getDamage()
Golem
- Image golem
+ <Create> Golem(double,Player)
+ int getZ()
+ void draw(GraphicsContext)
+ void update()
+ double getX()
+ double getY()
+ int getHP()
+ void setHP(int)
+ int getDamage()
Boss
- Image slimeRight
+ <Create> Boss(double,double,double,Player)
+ int getZ()
+ void draw(GraphicsContext)
+ void update()
+ double getX()
+ double getY()
+ int getHP()
+ void setHP(int)
+ int getDamage()


shareObject
IRenderable
+ int getZ()
+ void draw(GraphicsContext)
+ boolean isVisible()
RenderableHolder
- RenderableHolder instance
- List<IRenderable> renderables
- Comparator<IRenderable> comparator
+ <Create> RenderableHolder()
+ RenderableHolder getInstance()
+ void add(IRenderable)
+ void remove(IRenderable)
+ List<IRenderable> getEntities()
+ void reset()
item
BaseItem
- String name
- Rectangle soldArea
- double x
+ <Create> BaseItem(String)
+ Image getImage()
+ void useItem()
potion
manaPotion
- boolean drawn
- Image manaPotion
+ <Create> manaPotion()
+ int getZ()
+ void draw(GraphicsContext)
+ Image getImage()
+ void useItem()
powerPotion
- boolean drawn
- Image powerPotion
+ <Create> powerPotion()
+ int getZ()
+ void draw(GraphicsContext)
+ Image getImage()
+ void useItem()
healPotion
- boolean drawn
- Image healPotion
+ <Create> healPotion()
+ int getZ()
+ void draw(GraphicsContext)
+ Image getImage()
+ void useItem()
Key
- boolean drawn
- Image key
+ <Create> Key()
+ int getZ()
+ void draw(GraphicsContext)
+ Image getImage()
+ void useItem()
Broom
- boolean used
- boolean drawn
- Image broom
+ <Create> Broom()
+ int getZ()
+ void draw(GraphicsContext)
+ Image getImage()
+ void useItem()
+ boolean isUsed()
weapon
Shield
- boolean drawn
- Image shield
+ <Create> Shield()
+ int getZ()
+ void draw(GraphicsContext)
+ Image getImage()
+ void useItem()
Wand
- boolean drawn
- Image wand
+ <Create> Wand()
+ int getZ()
+ void draw(GraphicsContext)
+ Image getImage()
+ void useItem()


scene
GameScene
+ Scene getScene()
+ void listener()
+ void inventoryHandle(MouseEvent)
+ void gameloop()
SceneControl
- Stage stage
- HomeScene homeScene
- ItemScene gameScene
- MonsterScene monsterScene
- BossScene bossScene
+ <Create> SceneControl(Stage)
+ void showHomeScene()
+ void showItemScene()
+ void showMonsterScene()
+ void showBossScene()
FightScene
+ void attackOperation(KeyEvent)
+ void gameOver()
HomeScene
- MediaPlayer mediaPlayer
- SceneControl sceneControl
- Scene scene
+ <Create> HomeScene(SceneControl)
+ ImageView backgroundImageView()
+ Text gameName()
+ Button startButton()
+ VBox instructionBox()
+ Text instructionText()
+ Text startText()
+ Button howToPlayButton()
+ void mediaPlayer()
+ void translateTransition(Text)
+ Scene getScene()
ItemScene
- SceneControl sceneControl
- Scene scene
- ItemScene(Logic logic
- GameScreen gameScreen
- boolean sceneState
+ <Create> ItemScene(SceneControl)
+ void listener()
+ void inventoryHandle(MouseEvent)
+ void gameloop()
+ Scene getScene()
BossScene
- boolean coolDown
- StackPane root
- SceneControl sceneControl
- Scene scene
- BossScene(Logic logic
- FightScreen fightScreen
+ <Create> BossScene(SceneControl)
+ Scene getScene()
+ void listener()
+ void attackOperation(KeyEvent)
+ void inventoryHandle(MouseEvent)
+ void gameloop()
# void delayLost(AnimationTimer)
# void delayWin(AnimationTimer)
+ void gameOver()
+ void gameWin()
MonsterScene
- boolean coolDown
- StackPane root
- SceneControl sceneControl
- Scene scene
- MonsterScene(Logic logic
- FightScreen fightScreen
+ <Create> MonsterScene(SceneControl)
+ void listener()
+ void attackOperation(KeyEvent)
+ void inventoryHandle(MouseEvent)
+ void gameloop()
+ void delayLost(AnimationTimer)
+ void gameOver()
+ Scene getScene()

Implementation Detail

1. Package Drawing

1.1 Class GameScreen extends Canvas

This class is for drawing objects in the scene.

1.1.1 Constructor

Name	Description
+ GameScreen(double width, double height)	Set canvas Width and Height

1.1.2 Methods

Name	Description
+ paintComponent()	Draw entity that visible and IRenderable

1.2 Class FightScreen extends GameScreen

This class is for drawing objects in the scene. It is used in MonsterS stage and Boss stage (stages which have fighting events).

1.2.1 Constructor

Name	Description
+ FightScreen(double width, double height)	Set canvas Width and Height

1.2.2 Methods

Name	Description
+ paintComponent()	Draw entity that visible and IRenderable
+ drawPlayerHP(GraphicsContext gc, Player player)	Draw player's HP stack
+ drawPlayerMana(GraphicsContext gc, Player	Draw player's mana stack

player)	

2. Package logic

2.1 Package game

2.1.1 Class BaseSceneLogic

This is an abstract class.

2.1.1.1 Fields

Name	Description
# List<IRenderable> objectContainer	List of IRenderable objects in scene
# Chest chest	Represent the Chest

2.1.1.2 Constructor

Name	Description
+ BaseSceneLogic()	- Initialize objectContainer & chest.

2.1.1.3 Methods

Name	Description
# void addElement(IRenderable element)	Add element to objectContainer and RenderableHolder.*getInstance*()
+ abstract void logicUpdate()	Update SceneLogic
+ Chest getChest()	Getter of chest

2.1.2 Class FightSceneLogic extends BaseSceneLogic

This is an abstract class.

2.1.2.1 Fields

Name	Description
# ArrayList<BaseMonster> monsters	List of monsters in scene
# InventorySlot inventorySlot	Slot of items in chest
# ArrayList<Magic> magicList	List of magics in scene
# ArrayList<BaseItem> items	List of items in scene
# Player player	Represent the Player

2.1.2.2 Constructor

Name	Description

+ FightSceneLogic()	super();
	- Initialize monsters, magicList, monster map
	- Set up player, items, inventorySlot
	- Add all things in RenderableHolder.*getInstance*()
	- Call startManaRegeneration(player);
	startHpRegeneration(player);

2.1.2.3 Methods

Name	Description
+ void startManaRegeneration(Player player)	Increase player’s mana by 1 every 5 seconds
+ void startHpRegeneration(Player player)	Increase player’s HP by 1 every 5 seconds
+ void logicUpdate()	- Update location of monsters, player, magic

	- Check attack
# void addMonster(BaseMonster monster)	Add monster to monsters
+ abstract void reset()	Reset instance
+ Getters of all fields	-

2.1.3 Class ItemSceneLogic extends GameSceneLogic

This class is logic for item collecting stage.

2.1.3.1 Fields

Name	Description
- ItemSceneLogic *instance*	Represent scene's instance
- InventorySlot inventorySlot	Slot of item in chest
+ ArrayList<BaseItem> items	List of items in scene
- Player player	Represent the Player
- Broom broom	Represent the Broom

2.1.3.2 Constructor

Name	Description
+ ItemSceneLogic()	super(); - Initialize items, item map, player, InventorySlot - Set up player, items, inventorySlot - Add all things in RenderableHolder.*getInstance*() - Add object by addElement(IRenderable element) - Add item by addItem(BaseItem item)

2.1.3.3 Methods

Name	Description
- void addItem(BaseItem item)	Add item to items
+ void logicUpdate()	- Update player's position - Check player collide items
+ boolean sceneUpdate()	Check player exit scene
+ void reset()	Reset instance
+ Getters of all fields	-

2.1.4 Class MonsterSceneLogic extends FightSceneLogic

This class is the logic for the Monster fighting stage.

2.1.4.1 Fields

Name	Description
- MonsterSceneLogic *instance*	Represent scene's instance
- int generatedMonsterCount	Counter of generated monsters
- Bat bat	Represent Bat
- Golem golem	Represent Golem

2.1.4.2 Constructor

Name	Description
+ MonsterSceneLogic()	super(); - Initialize bat and golem - Add bat and golem by addElement(IRenderable element) addItem(BaseItem item)

2.1.4.3 Methods

Name	Description
- void startMonsterGeneration()	Generation new monsters in every 5 second for 5 times by generateRandomMonster()
- void generateRandomMonster()	Randomly generated monsters.
- void addMagic	- Initialize magic - Set magic state - Add magic by addElement(IRenderable element) - Add magic to magicList
+ void reset()	reset instance
+ MonsterSceneLogic getInstance()	Getters of instance
+ int getGeneratedMonsterCount	Getters of generatedMonsterCount

2.1.5 Class BossSceneLogic extends FightSceneLogic

This class is the logic for the Boss fighting stage.

2.1.5.1Fields

Name	Description
- BossSceneLogic *instance*	Represent scene's instance
- FireBomb fireBomb	Represent FireBomb

- Boss boss	Represent Boss

2.1.5.2Constructor

Name	Description
+ BossSceneLogic()	super(); - Initialize boss - Add boss by addElement(IRenderable element) addMonster(BaseMonster monster) - Call fireBombLoop()

2.1.5.3Methods

Name	Description
- void fireBombLoop()	- Initialize fireBomb every 5 second - Add fireBomd by addElement(fireBomb)
- void logicUpdate()	super.logicUpdate(); - Check player collide fireBomb
+ void reset()	Reset instance
+ BossSceneLogic getInstance()	Getter of instance

2.2 Package item

2.2.1 Abstract class BaseItem extends entity implement IRendable

This is the super class for other items.

2.2.1.1 Fields

Name	Description
+ String name	Name of item
+ Rectangle solidArea	It is used to check whether Item collides with other objects.
+ double x, y	Position of item

2.2.1.2 Constructor

Name	Description
+ BaseItem(String name)	Set name

2.1.2.3 Methods

Name	Description
+ abstract Image getImage()	Getter of
+ abstract void useItem ()	Add abilities to player

2.2.2 Class Broom extends BaseItem implement IRendable

2.2.2.1 Fields

Name	Description
- boolean used	Check broom is used
- boolean drawn	Check broom is drawn
- Image broom	Represent Broom image

2.2.2.2 Constructor

Name	Description
+ Broom()	- Set name to “broom” - Set used to false

2.2.2.3 Methods

Name	Description
+ int getZ()	- Return 0 - It is used to draw object on canvas in sequence
+ draw(GraphicsContext gc)	- Draw broom’s image in random position - Set solidArea
+ Image getImage()	Getter of image
+ void useItem ()	- Set player’s speed to 5 - Set used = true
+ boolean isUsed()	Getter of used

2.2.3 Class Key extends BaseItem implement IRendable

2.2.3.1 Fields

Name	Description
- boolean drawn	Check Key is drawn
- Image key	Represent Key image

2.2.3.2 Constructor

Name	Description

+ Key()	- Set name to “key”

2.2.3.3 Methods

Name	Description
+ Int getZ()	- Return 0 - It is used to draw object on canvas in sequence
+ draw(GraphicsContext gc)	- Draw key’s image in random position - Set solidArea
+ Image getImage()	Getter of image
+ void useItem ()	Not add special ability

2.3 Package potion

2.3.1 Class healPotion extends BaseItem implement IRendable

2.3.1.1Fields

Name	Description
- boolean drawn	Check healPotion is drawn
- Image healPotion	Represent healPotion image

2.3.1.2 Constructor

Name	Description
+ healPotion()	- Set name to “healPotion”

2.3.1.3 Methods

Name	Description
+ int getZ()	- Return 0 - It is used to draw object on canvas in sequence
+ draw(GraphicsContext gc)	- Draw healPotion's image in random position - Set solidArea
+ Image getImage()	Getter of image
+ void useItem ()	Increase player's HP by 5

2.3.2 Class manaPotion extends BaseItem implement IRendable

2.3.2.1 Fields

Name	Description
- boolean drawn	Check manaPotion is drawn
- Image manaPotion	Represent manaPotion image

2.3.2.2 Constructor

Name	Description
+ manaPotion()	- Set name to "manaPotion"

2.3.2.3 Methods

Name	Description
+ Int getZ()	- Return 0 - It is used to draw object on canvas in sequence
+ draw(GraphicsContext gc)	- Draw mana Potions image in random position - Set solidArea

+ Image getImage()	Getter of image
+ void useItem ()	Increase player's mana by 5

2.3.4 Class powerPotion extends BaseItem implement IRendable

2.3.4.1 Fields

Name	Description
- boolean drawn	Check powerPotion is drawn
- Image powerPotion	Represent powerPotion image

2.3.4.2 Constructor

Name	Description
+ powerPotion()	- Set name to "powerPotion"

2.3.4.3 Methods

Name	Description
+ int getZ()	- Return 0 - It is used to draw object on canvas in sequence
+ draw(GraphicsContext gc)	- Draw power Potions image in random position - Set solidArea

+ Image getImage()	Getter of image
+ void useItem ()	Increase player's damage by 2

2.4 Package weapon

2.4.1Class Shield extends BaseItem implement IRendable

2.4.1.1 Fields

Name	Description
- boolean drawn	Check Shield is drawn
- Image shield	Represent Shield image

2.4.1.2 Constructor

Name	Description
+ Shield()	- Set name to "shield"

2.4.1.3 Methods

Name	Description
+ int getZ()	- Return 0 - It is used to draw object on canvas in sequence
+ draw(GraphicsContext gc)	- Draw shield's image in random position - Set solidArea
+ Image getImage()	Getter of image

+ void useItem ()	player.setUsingShield(true)

2.4.2 Class Wand extends BaseItem implement IRendable

2.4.2.1 Fields

Name	Description
- boolean drawn	Check Wand is drawn
- Image wand	Represent Wand image

2.4.2.2 Constructor

Name	Description
+ Wand()	- Set name to “wand”

2.4.2.3 Methods

Name	Description
+ int getZ()	- Return 0 - It is used to draw object on canvas in sequence
+ draw(GraphicsContext gc)	- Draw wand’s image in random position - Set solidArea
+ Image getImage()	Getter of image
+ void useItem ()	Not add special ability

2.5 Package map

2.5.1 Class Chest extends Entity

2.5.1.1 Fields

Name	Description
- Rectangle solidArea	It is used to check whether Chest is Clicked
- Image chest	Represent Chest image

2.5.1.2 Constructor

Name	Description
+ Chest()	-

2.5.1.3 Methods

Name	Description
+ int getZ()	- Return 11 - It is used to draw object on canvas in sequence
+ draw(GraphicsContext gc)	- Draw chest's image - Set solidArea
+ Rectangle getSolidArea()	Getter of solidArea

2.5.2 Class Door extens Entity

2.5.2.1 Fields

Name	Description
- static Door *instance*	Represent Door's instance
- Rectangle doorArea	It is used to check whether Door collides with Player.
- final double x,y	Position of Door
- Image door	Represent Door image

2.5.2.2 Constructor

Name	Description
+Door()	- Set x to 400 - Set y to 550

2.5.3.3 Methods

Name	Description
+ int getZ()	- Return 11 - It is used to draw object on canvas in sequence
+ draw(GraphicsContext gc)	- Draw door's image - Set doorArea
+ Getters of instance , DoorArea	-

2.5.3 Class InventorySlot implement IRenderable

2.5.3.1 Fields

Name	Description
- ArrayList<Rectangle> slotAreaList	List of slot's area
- boolean visible	Check visible

2.5.3.2 Constructor

Name	Description
+InventorySlot()	- Set visible to false - Initailize slotAreaList

2.5.3.3 Methods

Name	Description
+ int getZ()	- Return 11 - It is used to draw object on canvas in sequence
+ draw(GraphicsContext gc)	- Draw background inventory slot - Set area and add to slotAreaList - Draw item's image
+ Setter of visible	-
+ Getters of visible, slotAreaList	-

2.5.4 Class ItemMap extends Entity

This is a class for item collecting stage's floor.

2.5.4.1 Fields

Name	Description
- Image castleSprite	Represent castleSprite image
- Image bookShelf	Represent bookShelf image
- Image window	Represent window image

2.5.4.2 Methods

Name	Description
+ int getZ()	- Return -100 - It is used to draw object on canvas in sequence
+ draw(GraphicsContext gc)	- Draw castleSprite's image - Draw bookShelf's image - Draw window's image

2.5.5 Class MonsterMap extends Entity

This is a class for monster fighting stage's floor.

2.5.5.1 Fields

Name	Description
- Image grass	Represent grassSprite image
- Image flower	Represent flower image

2.5.5.2 Methods

Name	Description
+ int getZ()	- Return -100 - It is used to draw object on canvas in sequence
+ draw(GraphicsContext gc)	- Draw grassSprite's image - Draw flower's image

2.5.6 Class Wall extends Entity

Fields

Name	Description
- Image wall	Represent Wall image

Methods

Name	Description
+ int getZ()	- Return 10 - It is used to draw object on canvas in sequence
+ draw(GraphicsContext gc)	- Draw wall's image

2.6 Package monsters

2.6.1 Class BaseMonster extends Entity implements Attackable

This is an abstract class for other monsters.

2.6.1.1 Fields

Name	Description
+ String name	Name of monster
+ double x,y	Position of monster
+ double speed	Speed of monster
+ Player player	Represent Player
+ Rectangle solidArea	It is used to check whether Monster collides with Player.
+ int HP	HP of monster
+ int damage	damage of monster

2.6.1.2 Constructor

Name	Description
+ BaseMonster(String name)	Set name

2.6.1.3 Methods

Name	Description
+ int getZ()	- Return 0 - It is used to draw object on canvas in sequence
+ draw(GraphicsContext gc)	- Draw monster's image - Set solidArea
+ void update()	Update monster's location

+ Setter of HP	-
+ Getters of Damage, HP	-

2.6.2 Class Bat extends BaseMonster

2.6.2.1 Fields

Name	Description
- Image bat	Represent bat image

2.6.2.2 Constructor

Name	Description
+ Bat(double speed, Player player)	- Set name to “bat” - Random x, y - Set bat.speed to speed - Set bat.player to player - Set HP to 2 - Set damage to 1

2.6.2.3 Methods

Name	Description
+ int getZ()	- Return 0 - It is used to draw object on canvas in sequence
+ draw(GraphicsContext gc)	- Draw bat’s image in random position
+ void update()	- Update Bat’s position (Bat always move to player) - Update solidArea

+ Setter of HP	-
+ Getters of Damage, HP	-

2.6.3 Class Boss extends BaseMonster

2.6.3.1 Fields

Name	Description
- Image slimeRight	Represent Boss image

2.6.3.2 Constructor

Name	Description
+ Boss(double x, double y, double speed, Player player)	- Set name to “Boss” - Set position to x,y - Set boss.speed to speed - Set boss.player to player

2.6.3.3 Methods

Name	Description
+ int getZ()	-return 0 -It is used to draw object on canvas in sequence
+ draw(GraphicsContext gc)	- Draw boss’s image in position x, y
+ void update()	- Update Boss’s position (Boss always move to player) - Update solidArea
+ Setter of HP	-
+ Getters of Damage, HP	-

2.6.4 Class FireBomb implements IRenderable

2.6.4.1 Fields

Name	Description
- double x, y	Position of FireBomb
- boolean visible	Check visible
+ Player player	Represent Player
+ Rectangle solidArea	It is used to check whether FireBomb collides with Player.
+ int damage	Damage of FireBomb
- Image fireBomb	Represent FireBomb image

2.6.4.2 Constructor

Name	Description
+ FireBomb(Player player)	- Set visible to true - Set player - Random x, y - Set damage to 2 - Call cycle();

2.6.4.3 Methods

Name	Description
+ int getZ()	-return 1

	-It is used to draw object on canvas in sequence
+ draw(GraphicsContext gc)	- Draw fireBomb's image in random position - Set solidArea
+ void cycle()	Set visible to false every 1 second
+ boolean isVisible()	Getter of visible

2.6.5 Class Golem extends BaseMonster

2.6.5.1 Fields

Name	Description
- Image golem	Represent Golem image

2.6.5.2 Constructor

Name	Description
+ Golem(double speed, Player player)	- Set name to "golem" - Random x, y - Set golem.speed to speed - Set golem.player to player - Set HP to 5 - Set damage to 2

2.6.5.3 Methods

Name	Description
+ int getZ()	- return 0 - It is used to draw object on canvas in sequence

+ draw(GraphicsContext gc)	- Draw golem's image in random position
+ void update()	- Update Golem's position (Golem always move to player) - Update solidArea
+ Setter of HP	-
+ Getters of Damage, HP	-

2.7 Package player

2.7.1 Class Magic extend Entity

This is a class for magic which player shoot to attack monsters.

2.7.1.1 Fields

Name	Description
+ Rectangle solidArea	It is used to check whether Magic collides with other objects.
- WalkState state	Determine Magic moving direction
- Player player	Represent the player
- double x,y	Represent the Magic Position

2.7.1.2Constructor

Name	Description
+ Magic()	- Set up magic - Initialize player field to player which is initialized in ItemSceneLogic, MonsterSceneLogic and BossSceneLogic (it is the same object in these 3

	classes)

2.7.1.3 Methods

Name	Description
+ void update()	Updating the magic’s position
+ void setMagicState()	- Setting the magic state which is referred from the player's WalkState - It determines the magic’s moving direction.
+ int getZ()	Return 0
+ void draw(GraphicContext gc)	Draw a magic ball image
+ void updateSolidArea()	Update Magic’s solidArea
+ double getX()	Getter for x
+ double getY()	Getter for y

2.7.2 Class Player extend Entity

2.7.2.1 Fields

Name	Description
+ double velocityX	Velocity of player in x-axis
+ double velocityY	Velocity of player in y-axis
- boolean weapon	Check whether player has weapon (Wand)

- double x,y	Player’s position in x-axis and y-axis
- static final int WALK	Constant for changing Player’s position
- int speed	Player’s speed
+ Rectangle solidArea	It is used to check whether Player collides with other objects.
- WalkState walkState	Represents player walking direction
+ ArrayList<BaseItem> playerItem	Represents Player’s Items in Inventory
- final int MAX_HP	-Max_HP = 20 -Player’s max hp
- final int MAX_MANA	-Max_MANA = 20 -Player’s max mana
- int HP	Player’s current hp
- int mana	Player’s current mana
- int damage	Damage which Player can make while attack to monster
- boolean gameOver	-Represents state whether it is over yet -It is initially set as false
- boolean isUsingShield	-Represents whether Player is using Shield -It is initially set as false -If its value is true, the player can’t be attacked.
- Timeline shieldTimer	Period that player is using Shield
- Timeline attackedTimer	Period when the player cannot receive damage,it starts counting after the player receives damage.
- boolean canBeAttacked	Representing whether the player can receive damage, it works together with attackedTimer.
- Image playerFront	Represent playerFront image
- Image playerBack	Represent playerBack image

- Image playerLeft	Represent playerLeft image
- Image playerRight	Represent playerRight image
- Image broomFront	Represent broomFront image
- Image broomBack	Represent broomBack image
- Image broomLeft	Represent broomLeft image
- Image broomRight	Represent broomRight image

2.7.2.2 Constructor

Name	Description
+ player()	Set up Player

2.7.2.3 Methods

Name	Description
+ void moveUpward()	Player moves upward.
+ void moveDownward()	Player moves downward.
+ void moveLeft()	Player moves left.
+ void moveRight()	Player moves right.
+ void update()	- Check KeyCode in Input Class and makes player walk in the preferred direction - Set player WalkState
+ void setPosition(double x, double y)	Set Player's position in the beginning of the scene
+ void setSpeed(int speed)	Set player's speed
+ WalkState getWalkState()	Getter for Player's walkState

+ WalkState setWalkState()	Setter for Player's walkState
+ boolean hasWeapon()	- Getter for weapon, check whether player has the weapon (Wand) - If it returns false,the player can't attack.
+ void checkWeapon()	- For loop check player's inventory whether player has weapon
+ int getZ	- Return 1 - It is used to draw object on canvas in sequence
+ void draw(GraphicContext gc)	- Draw player on canvas - Draw different dimension of player by checking WalkState - Draw player when player uses broom - Draw player's shadow when player use broom
+ void checkCollisionItem(ArrayList<BaseItem> items)	- It loop checks whether a player collides with BaseItem and remove that item from the canvas
+ boolean checkExitScene()	It checks whether the player collides to the door and has Key in ItemScene.If true Player goes to next scene.
+ void pickUpItem(BaseItem item)	- If player collides to item, it add item to player's inventory - It is used in checkCollisionItem
+ boolean isGameOver()	Getter for gameOver
+ void setGameOver(boolean gameOver)	Setter for gameOver
+ void getAttacked(ArrayList <BaseMonster> monsters)	- Check whether player is attacked by monster - Set player's position bounce back if player is attacked
+ void Attack(ArrayList <BaseMonster> monsters)	Player cast the magic
+ void checkMagicCollisionMonster (ArrayList<BaseMonster> monsters)	It checks whether the magic which the player cast collides with monster.If true, reduces

	monster’s hp and sets monster bounce back.
+ static void dropItem(double x, double y)	Drop items from monster if monsters are killed
+ boolean isCanBeAttacked()	Getter for canBeAttacked
+ void setCanBeAttacked(boolean canBeAttacked)	Setter for canBeAttacked
+ void coolDownDuration()	- Cool down for canBeAttacked - Start cool down when player received damage
+ void setUsingShield(boolean usingShield)	Setter of usingShield
+ boolean isUsingShield()	Getter of usingShield
+ int getMAX_HP()	Getter of MAX_HP
+ int getMAX_MANA()	Getter of MAX_MANA
+ int getHP()	Getter of hp (player’s current hp)
+ void setHP(int HP)	Setter of hp (player’s current hp)
+ int getMana()	Getter of mana (player’s current mana)
+ void setMana(int mana)	Setter of mana (player’s current mana)
+ int getDamage()	Getter of damage (damage which player can make)
+ void setDamage(int damage)	Setter of damage (damage which player can make)
+ double getX()	Getter of player’s position in x-axis
+ double getY()	Getter of player’s position in y-axis
+ ArrayList<BaseItem> getPlayerItem()	Getter of playerItem (player’s items in inventory)

2.8 Enum WalkState

This enum refers to the player moving direction. It is utilized in drawing different player's images which relate to the WalkState. For example, while the player is moving to the left, the WalkState is LEFT.Draw method gets that WalkState and draws an image of the player moving in the left.

- Enum

Name	Description
UP	Player moving up in y-axis
DOWN	Player moving down in y-axis
LEFT	Player moving to the left
RIGHT	Player moving to the right

2.9 Interface Attackable

Name	Description
# int getHP();	Getter for object's HP
# void setHP(int hp) ;	Setter for object's HP
# int getDamage();	Getter for damage that object's can make

2.10 Class Entity implement IRenderable

2.10.1 Fields

Name	Description
- int z	Refers to z which is used to compare with other object and draw in sequence
# boolean visible	Indicates whether that object is visible on the canvas

2.10.2 Constructor

Name	Description
+ Entitiy()	Initially sets visible as true

2.10.3 Methods

Name	Description
+ boolean isVisible	Getter for visible
+ int getZ()	Getter for z

3. Package scene

3.1 Interface GameScene

This interface is implemented in scenes which are in the game.

3.1.1 Methods

Name	Description
+ Scene getScene()	Getter for each scene
+ void listener()	Handle the input from keyboard and mouse
+ void inventoryHandle(MouseEvent mouseEvent)	Handle when inventory slot is triggered
+ void gameloop()	Manages game flow

3.2 Interface FightScene

This interface is implemented in the scenes which have attacking events.

3.2.1 methods

Name	Description
+ void attackOperation(KeyEvent event)	Handle when player attack and cooldown player's attack
+ void gameOver()	Handle when the game is over

3.3 Class SceneControl

This class is for managing the scenes in this application.

3.3.1 Fields

Name	Description
- Stage stage	Refers to stage
- HomeScene homeScene	Refers to HomeScene

- ItemScene itemScene	Refers to itemScene
- MonsterScene monsterScene	Refers to MonsterScene
- BossScene bossScene	Refers to BossScene

3.3.2 Constructor

Name	Description
+ SceneControl (Stage stage)	Set this.stage as argument

3.3.3 Methods

Name	Description
+ void showHomeScene()	- Initialize homeScene - Set stage's scene as homeScene
+ void showItemScene()	- Initialize itemScene - Set stage's scene as itemScene
+ void showMonsterScene()	- Initialize monsterScene - Set stage's scene as monsterScene
+ void showBossScene()	- Initialize bossScene - Set stage's scene as bossScene

3.4 HomeScene

This is the game's main menu scene.

3.4.1 Fields

Name	Description
+ static MediaPlayer mediaPlayer	Background music
- SceneControl sceneControl	Represents to SceneControl class

- Scene scene	Represents to this scene

3.4.2 Constructor

Name	Description
+ HomeScene (SceneControl sceneControl)	Initialize scene, nodes and mediaPlayer

3.4.3 Methods

Name	Description
- ImageView backgroundImageView()	This scene’s background image.
- Text gameName()	Game name’s Text
- Button startButton()	Button to start the game
- VBox instructionBox()	How to play’s page
- Text instructionText()	Explain how to play
- Text startText()	Invite player to play
- Button howToPlayButton()	Button which manages how to play page
- void mediaPlayer()	Manages and plays background music
- void translateTransition(Text gameName)	Makes gameName moving wavily
+ Scene getScene()	Getter for this class

3.5 class ItemScene implements GameScene

This Class manages the item collecting stage.

3.5.1 Fields

Name	Description

- SceneControl sceneControl	Represent to SceneControl
- Scene scene	Represents to this scene
- ItemSceneLogic logic	Represents to class ItemSceneLogic
- GameScreen gameScreen	Represents to GameScreen which is ItemScene's Canvas
- boolean sceneState	It refers whether this scene end

3.5.2 Constructor

Name	Description
+ ItemScene (SceneControl sceneControl)	Set up itemScene

3.5.3 Methods

Name	Description
+ void listener()	Receives input from mouse and keyboard.
+ void inventoryHandle(MouseEvent mousevent)	Handles when chest icon is triggered and manages about inventory slot
+ void gameloop()	Manage ItemScene game loop by using AnimationTimer
+ Scene getScene()	Getter for ItemScene

3.6 Class MonsterScene implements GameScene, FightScene

This class manages Monster fighting scene (First stage).

3.6.1 Fields

Name	Description

- boolean coolDown	Cool down for the player's attack.
- StackPane root	Refers to root node of this scene graph
- SceneControl sceneControl	Represents SceneControl
- Scene scene	Represents to this scene
- MonsterSceneLogic logic	Represents MonsterSceneLogic
- FightScreen fightScreen	Represents FightScreen which is MonsterScene's Canvas.

3.6.2 Constructor

Name	Description
+ MonsterScene(SceneControl sceneControl)	Set up MonsterScene

3.6.3 Methods

Name	Description
+ void listener()	Receives input from mouse and keyboard.
- void attackOperation(KeyEvent keyEvent)	-Handle about player attack -It uses Thread to cool down after the player's attacking to prevent too frequently attacking.
+ void inventoryHandle(MouseEvent mousevent)	Handles when chest icon is triggered and manages about inventory slot
+ void gameloop()	Manage MonsterScene game loop by using

	AnimationTimer
+ void delayLost(AnimationTimer animationTimer)	-Delay after the player dies for game's smoothly. -Add sound effect
+ void gameOver()	Handle when the game is over show the text and button which can switch to the other scenes.
+ Scene getScene()	Getter for MonsterScene

3.7 class BossScene implements GameScene, FightScene

This Class manages the boss fighting stage.

3.7.1 Fields

Name	Description
- boolean coolDown	Cool down for the player's attack.
- StackPane root	Refers to root node of this scene graph
- SceneControl sceneControl	Represents SceneControl
- Scene scene	Represents to this scene
- BossSceneLogic logic	Represents BossSceneLogic
- FightScreen fightScreen	Represents FightScreen which is BossScene's Canvas.

3.7.2 Constructor

Name	Description
+ BossScene(SceneControl sceneControl)	set up BossScene

3.7.3 Methods

Name	Description
+ void listener()	Receives input from mouse and keyboard.
- void attackOperation(KeyEvent keyEvent)	- Handle about player attack - It uses Thread to cool down after the player's attacking to prevent too frequently attacking.
+ void inventoryHandle(MouseEvent mousevent)	Handles when chest icon is triggered and manages about inventory slot
+ void gameloop()	Manage MonsterScene game loop by using AnimationTimer
+ void delayLost(AnimationTimer animationTimer)	-Delay after the player dies for game's smoothly. -Add sound effect
+ void delayWin(AnimationTimer animationTimer)	Delay after the player conquers the boss for game's smoothly.
+ void gameOver()	Handle when the game is over (In case player loss). It shows the text and button which can switch to the other scenes.
+ void gameWin()	Handle when the player wins the game.It shows the text and button which can switch to the other scenes.
+ Scene getScene()	Getter for MonsterScene

4. Package utils

4.1 Class Config

This class contains the constants for some value.

4.1.1 Fields

Name	Description
+ static double sceneWidth	sceneWidth = 800;
+ static double sceneHeight	sceneHeight = 400;
+ static double playerWidth	playerWidth = 50;
+ static double playerHeight	playerHeight = 93;
+ static double playerWidthWBroom	- player's width with broom - playerWidthWBroom = 82;

4.2 Class Input

This class helps manage key code from keyboard's input.

4.2.1 Fields

Name	Description
- static ArrayList<KeyCode> keyPressed = new ArrayList<>()	Initialize the ArrayList which contains input keys from the keyboard.

4.2.2 Methods

Name	Description
+ static boolean getKeyPressed(KeyCode	Checking whether keyPressed contains the

keyCode)	keyCode
+ static void setKeyPressed(KeyCode keyCode, boolean pressed)	Add input from keyboard to ArrayList<KeyCode> keyPressed
+ static ArrayList<KeyCode> getKeyPressedList()	Getter for ArrayList<KeyCode> keyPressed

5. Package sharedObject

5.1 Interface IRenderable

This interface is implemented in Class which requires drawing an object on canvas.

3.1.1 Methods

Name	Description
+ int getZ()	Return Z, it is used to compare and draw objects in order.
+ void draw(GraphicContext gc)	Draw objects on the canvas.
+ boolean isVisible()	Return boolean whether the object is visible.

5.2 Class RenderableHolder

This class helps manage drawing an object on canvas.

5.2.1 Fields

Name	Description
- static final RenderableHolder instance = new RenderableHolder()	Make this class singleton during a game (it is reset after finish the game)
- List<IRenderable> entities	List that contains entity to draw on canvas
- Comparator<Irenderable> comparator	the comparator of Z for drawing in order on canvas

5.2.2 Constructor

Name	Description
+ RenderableHolder()	-initialize entities -initialize comparator by compare z-axis of each object

5.2.3 Methods

Name	Description
+ void add(IRenderable entity)	-add entity to entities -sorting entities by comparator
+ void remove (IRenderable entity)	Remove the entity from entities
+ List<IRenderable> getEntities()	getter for entities
+ void reset()	clear object in entities

6. Class Main extends Application

This is this project's main class. It initializes the application.

6.1 Constructor

Name	Description
+ static void main(String[] args)	It’s main constructor

6.2 Method

Name	Description
+ void start(Stage stage)	Set up JavaFX application