The Legend of Mushroom Documentation

Created by

Teerapat Chantaramanee 6430177021

Supawit Saeliew 6430392521

2110215 Programming Methodology

Semester 1 Year 2022

Chulalongkorn University

The Legend of Mushroom

Introduction

The Legend of Mushroom is inspired by the famous game, The Legend of Zelda. and Super Mario. The player must fight off the enemies to reach the Princess.

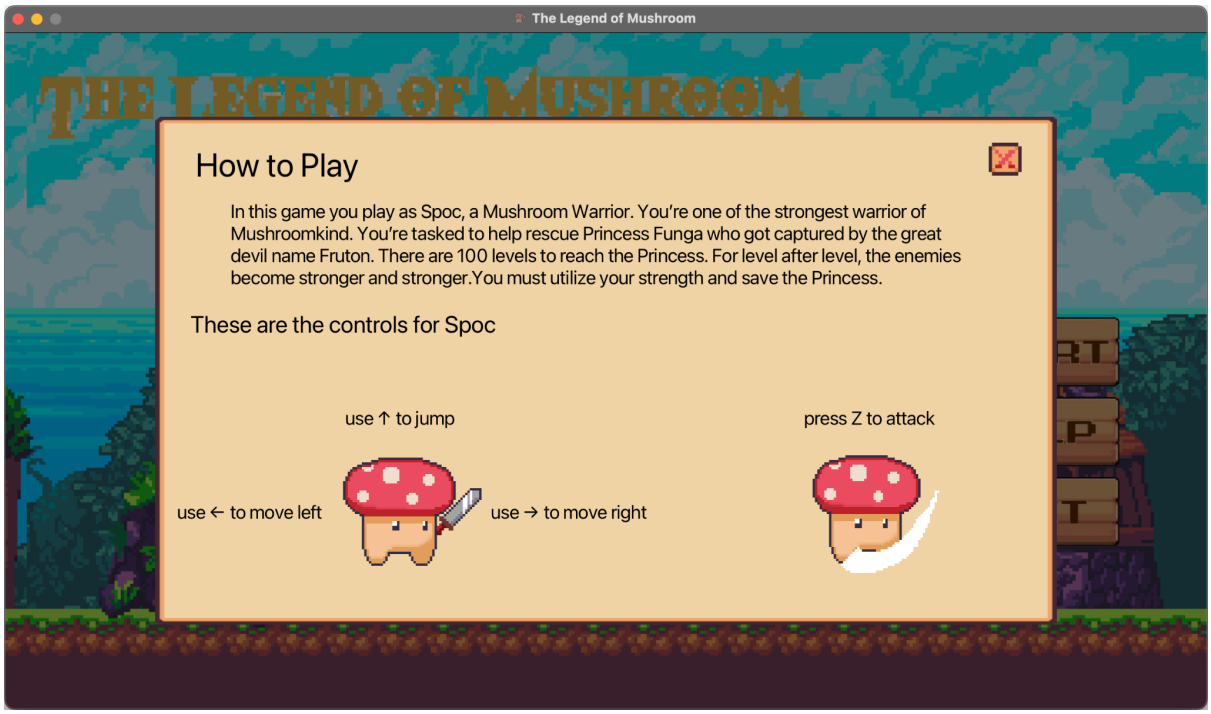
How to Play

Players use left and right arrow keys to move the character left and right, respectively. Use the up arrow key to jump and use the Z key to attack the monsters. To proceed to the next level, players must kill all enemies in the level that players are in. And walk to the right edge of the map to proceed. Note that players cannot proceed to the next level if there is(are) (a) monster(s) left in that level. When the players spawn in the next level. Players' health will be regenerated.

Main Menu

You can mute the music by clicking the sound button.

Help Pane



Gameplay

Pause Pane

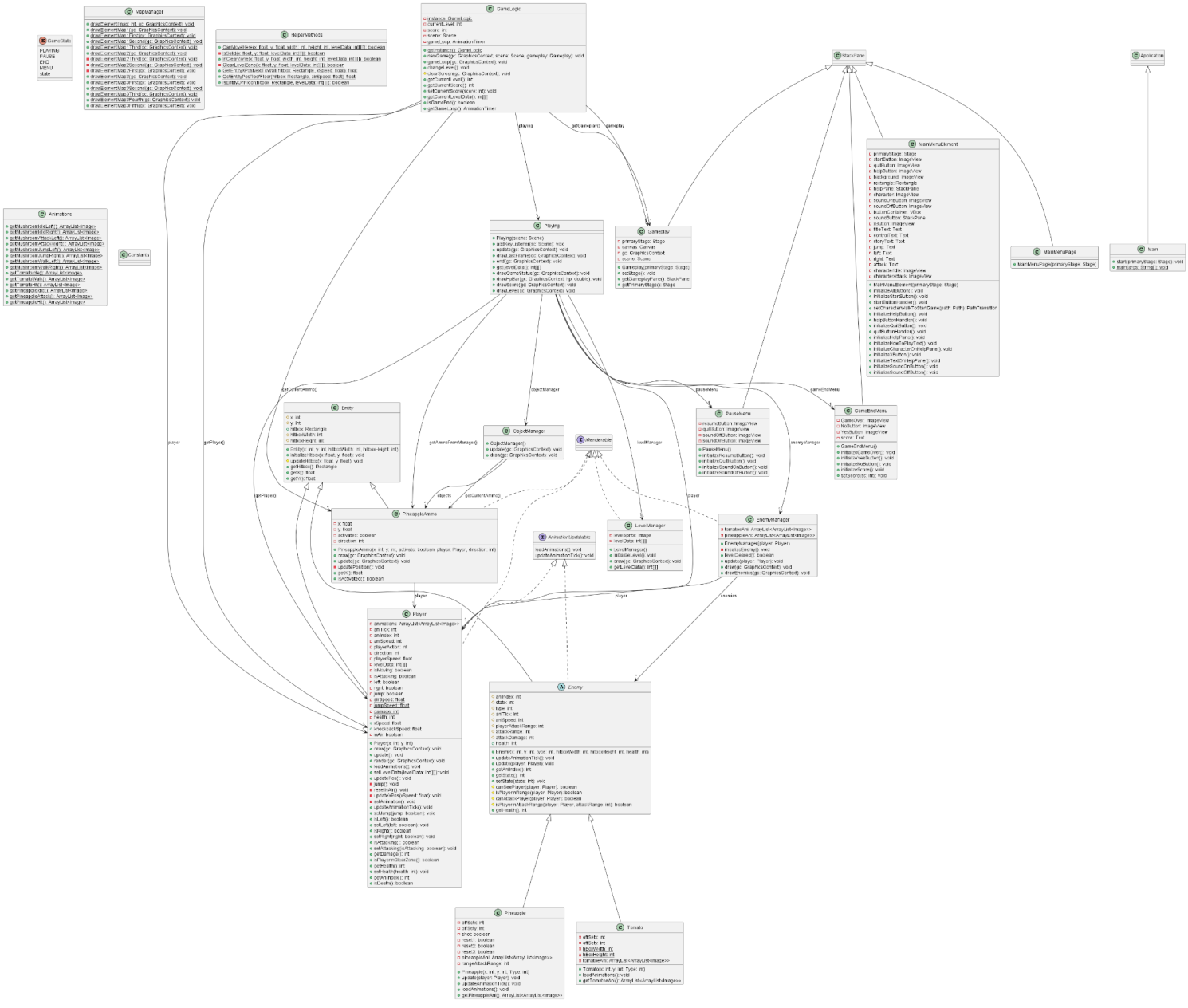
Game Over Pane

If you click yes, the game will reset to level 1 and reset score to 0.

else if you click no, you will return to the main menu.

Health bar, Level Indicator, and Score

Class Diagram



1.Package entities

1.1 Interface AnimationUpdatable

1.1.1 Methods

Name	Description
+ void loadAnimations()	Load animations for each Entity
+ void updateAnimationTick()	Update animations for each Entity

1.2 Abstract Class Entity

1.2.1 Fields

Name	Description
# float x	Coordinate of x in Pane
# float y	Coordinate of y in Pane
+ Rectangle hitbox	Hitbox for each Entity
# int hitboxWidth	Width of hitbox
# int hitboxHeight	Height of hitbox

1.2.2 Constructor

Name	Description
+ Entity(float x, float y, int hitboxWidth, int hitboxHeight)	Set entity position in pane. Set hitbox size

1.2.3 Methods

Name	Description
+ void initializeHitbox(float x, float y)	Initialize the hitbox of entity

# void updateHitbox(float x, float y)	Set position (x and y) of hitbox
+ Rectangle getHitbox()	Return the hitbox
+ int getX()	Return X coordinate
+ int getY()	Return Y coordinate

1.3 Abstract Class Enemy extends Entity

1.3.1 Fields

Name	Description
# int aniIndex	Current animation Index
# int state	Current state
# int type	Enemy's type
# int aniTick	Value to determine when to change animation
# int aniSpeed	Animation speed
# int playerAttackRange	Player's attack range
# int attackRange	Enemy's attack range
# int attackDamage	Enemy's attack damage
+ int health	Enemy's health

1.3.2 Constructor

+ Enemy (float x, float y, int Type, int hitboxWidth, int hitboxHeight, int health)	Set position, hitbox size, and health of the enemy

1.3.3 Methods

Name	Description

+ void updateAnimationTick()	Update the animation of each enemy state
# boolean canSeePlayer (Player player)	Check whether the player is in the vision range of this enemy. return true if it does. return false otherwise.
# boolean isPlayerInRange (Player player)	return true if player's x hitbox coordinate is less than or equal to attackRange of this enemy.
# boolean canAttackPlayer (Player player)	Check whether the player is in the attack range of this enemy in both X and Y axis. return true if it does. return false otherwise.
# boolean isPlayerInAttackRangeX (Player player)	Check whether the player is in this enemy X axis attack range. return true if it does. return false otherwise.
+ void update(Player player)	Check if enemy can see player and player attack its, change its state to HIT and decrease health by player's damage Check if enemy can attack player, decrease player's health by enemy's attack damage, and knockback the player Update the enemy's animations
+ int getAniIndex()	return aniIndex
+ int getState()	return state
+ int getHealth()	return health
+ void setState(int state)	set state

1.4 Class Player extends Entity implements IRenderable, AnimationUpdatable

1.4.1 Fields

Name	Description
- ArrayList<ArrayList<Image>> animations	List of player's animations
- int aniTick	Value to determine when to change animation
- int aniIndex	Current animation index
- int playerAction	Current player's state (Idle, jump, attack, walk)
- int direction	Current player direction
- float playerSpeed	Player's walk speed
- int [][] levelData	Store tile data. The value of Red of every pixel in dataImage refers to the index of tile in levelSprite.
- boolean isMoving	Check if player is moving
- boolean isAttaking	Check if player is attacking
- boolean left	Check if player direction equal left
- boolean right	Check if player direction equal right
- boolean jump	Check if player jump
- float airSpeed	Vertical speed of the player
- float jumpSpeed	Player's jump speed
- int damage	Player's attack damage
+ knockbackSpeed	Player's knockback speed
- boolean inAir	Check if player in the air

1.4.2 Constructor

Name	Description
+ Player (int x, int y)	Set player's position, initialize hitbox and load player's animations

1.4.3 Methods

Name	Description
+ void draw(GraphicContext gc)	Draw player by calling update() and render(gc)
+ void update()	Update the player : knockback, set inAir Call updatePos, updateAnimationTick, setAnimation and updateHitbox
+ render (GraphicsContext gc)	Draw player with current action
+ void loadAnimations()	Load all player's animations
+ void updateAnimationTick()	Update player's animations
+ void updatePos()	Update player's coordinate (x,y) including gravity)
- void jump()	Check if not inAir, set inAir equal true and airSpeed equal jumpSpeed
- void resetInAir()	Set inAir equal false, and airSpeed equal 0
- void updateXPos(float xSpeed)	Update player's x coordinate
- void setAnimation()	Set player's animation
+ void updateAnimationTick()	Update player's animation and play animation sound
+ void setLevelData(int[][] levelData)	Set levelData

+ void setJump(boolean jump)	Set jump
+ void setAttacking(boolean attacking)	Set isAttacking
+ void setLeft()	Set left
+ void setHealth(int health)	Set health
+ boolean isLeft()	Check if left
+ void setRight()	Set right
+ boolean isRight()	Check if right
+ boolean isAttacking	Check if player attacking
+ int getDamage()	return player's damage
+ int getHealth()	return player's health
+ int getAniIndex()	return AniIndex
+ boolean isDeath	return true if health less than or equal to 0
+ isPlayerInClearZone()	return true if InClearZone (method) equal true

1.5 Class Tomato extends Enemy

1.5.1 Fields

Name	**Description**
- ArrayList<ArrayList<Image>> tomatoeAni	List of tomatoes' animations
- int offSetx	Offset X coordinate

- int offSety	Offset Y coordinate
- int hBoxWidth	Hitbox width
- int hBoxHeight	Hitbox height

1.5.2 Constructor

Name	Description
+ Tomato(int x, int y, int Type)	Set type, health Initialize hitbox Call loadAnimations

1.5.3 Methods

Name	Description
+ void loadAnimations()	Load tomatoes' animations
+ ArrayList<ArrayList<Image>> getTomatoAni	Return all tomatoes' animations

1.6 Class Pineapple extends Enemy

1.6.1 Fields

Name	Description
- int offSetX	Offset X coordinate
- int offSetY	Offset Y coordinate
- boolean shot	Check if shot
- boolean reset1	Check if one animation cycle has passed
- boolean reset2	Check if two animation cycle has passed

- boolean reset3	Check if four animation cycle has passed
- ArrayList<ArrayList<Image>> pineappleAni	ArrayList to Store Animation of enemy pineapple
- rangeAttackRange	Pineapple's Attack range

1.6.2 Constructor

Name	**Description**
+ Pineapple(int x, int y, int Type)	Set type, health Initialize hitbox Call loadAnimations

1.6.3 Methods

+ void update(Player player)	
+ void updateAnimationTick()	Update pineapple animations
+ void loadAnimations()	Load pineapple animations
+ ArrayList<ArrayList<Image>> getPineappleAni()	Return all pineapple's animations

2.Package gamestates

2.1 Enum GameState

2.1.1 Enum

PLAYING, PAUSE,END, MENU

2.1.2 Fields

Name	Description
+ state	Current gamestate

2.2 Class Playing

2.2.1 Fields

Name	Description
- Player player	Current player
- LevelManager leveManager	For manager the maps
- EnemyManager enemyManager	For manager the enemies
- ObjectManager objectManager	For manager the objects
- PauseMenu pauseMenu	Pause pane
- GameEndMenu gameEnd	Game end pane

2.2.2 Constructor

Name	Description
+ Playing(Scene scene)	Initialize all fields Set levelData Start BattleMusic Add keyListinter

2.2.3 Methods

Name	Description
+ void addKeyListener(Scene sc)	Add key listener: escape = pause ⟶ = setRight ⟵ = setLeft ↑ = setJump z = attack
+ void update(GraphicsContext gc)	Update player, enemies, map, objects, game status
+ void drawLastFrame(GraphicsContext gc)	Stop update and show the pause menu
+ void end(GraphicsContext gc)	Stop update and show the end menu
+ void drawGameStatus(GraphicsContext gc)	Draw all game status by calling drawHpBar,drawScore, drawLevel
+ void drawHpBar(GraphicsContext gc,double hp)	Draw current player's health showing by green rectangle
+ void drawScore(GraphicsContext gc)	Draw current scores
+ void drawLevel(GraphicsContext gc)	Draw current level
+ Pleyer getPlayer()	Return current player
+ ArrayList<PineappleAmmo> getAmmoFromManager()	Return current pineapple's ammo
+ int[][] getLevelData()	Return current levelData

3.Package gui.element

3.1 Class GameEndMenu extends StackPane

3.1.1 Fields

Name	Description
- ImageView GameOver	Game over button.
- ImageView NoButton	No button.
- ImageView YesButton	Yes button.
- Text score	Text displaying score.

3.1.2 Constructor

Name	Description
+ GameEndMenu()	Initialize all fields by calling other methods

3.1.3 Methods

Name	Description
+ void initializeGameOver()	Initialize gameover board image
+ void initializeYesButton()	Initialize yes button Add hover effect. Add handler (Start new game).
+ void initializeNoButton()	Initialize no button Add hover effect. Add handler (Quit the game).
+ void initializeScore()	Initialize score text
+ void setScore(int sc)	Set score text to current score

3.2 Class MainMenuElement extends StackPane

3.2.1 Fields

Name	Description
- Stage primaryStage	Current Stage
- ImageView startButton	Start Button
- ImageView quitButton	Quit Button
- ImageView helpButton	Help Button
- ImageView background	Background Image
- Rectangle rectangle	Shadow for background
- StackPane helpPane	How to play and help pane
- ImageView character	Character image in Main Menu
- ImageView soundOnButton	Toggle sound on button
- ImageView soundOffButton	Toggle sound off button
- VBox buttonContainer	Group button
- StackPane soundButton	Sound button container
- ImageView xButton	close button for help pane
- Text titleText	Title Text in help pane
- Text controlText	Control Text in help pane
- Text storyText	Story Text in help pane
- Text jump	jump control text
- Text left	left control text
- Text right	right control text
- Text attack	attack control text

- ImageView characterIdle	Idling Character image in help pane
- ImageView characterAttack	Attacking Character image in help pane

3.2.2 Contructor

Name	Description
+ MainMenuElement(Stage primaryStage)	Initialize all fields Call initializeAllButton() method. Add node to the root

3.2.3 Methods

Name	Description
+ void initializeAllButton()	Initialize all button nodes by calling all initializing methods. After initializing them, add them to the container.
+ void initializeStartButton()	Initialize Start Button. Add hover effect. Add handler (Using startButtonHandler() Method).
+ void startButtonHandler()	Handle the startButton by calling setCharacterWalkToStartGame() method and play the mushroom walking animation in the background. Create new Gameplay.
+ PathTransition setCharacterWalkToStartGame(Path	Generate a linear path for the character to move.

path)	
+ initializeHelpButton()	Initialize Help Button. Add hover effect. Add handler (Using helpButtonHandler() Method).
+ void helpButtonHandler()	Display help pane.
+ void initializeQuitButton()	Initialize Quit Button. Add hover effect. Add handler (Using quitButtonHandler() Method).
+ void quitButtonHandler()	Exit the application
+ void initializeHelpPane()	Create the help pane. Add elements to it.
+ void initializeHowToPlayText()	Add text to help pane.
+ void initializeCharacterOnHelpPane()	Add idle and attack character to help pane
+ void initializeXButton()	Initialize the close button in the help pane.
+ void initializeTextOnHelpPane()	Add Description text to the help pane.
+ void initializeSoundOnButton()	Initialize sound-on button. add handler (Press this button and the music will not be muted).
+ void initializeSoundOffButton()	Initialize sound-off button. add handler (Press this button and the music will be muted).

3.3 Class PauseMenu

3.3.1 Fields

Name	Description
- ImageView resumeButton	Resume game button
- ImageView quitButton	Quit game button
- ImageView soundOffButton	Toggle Sound off button
- ImageView soundOnButton	Toggle Sound on button

3.3.2 Constructor

Name	Description
+ PauseMenu()	Call another initializer method.

3.3.3 Methods

Name	Description
+ void initializeResumeButton()	Initialize Resume Game Button. Add hover effect. Add handler (Resume the game).
+ void initializeQuitButton()	Initialize Quit Game Button. Add hover effect. Add handler (Quit the game).
+ void intializeSoundOnButton()	Initialize Sound On Button. Add hover effect. Add handler (Press the button and the music will not be muted).
+ void initializeSoundOffButton()	Initialize Sound Off Button. Add hover effect. Add handler (Press the button and the music will be muted).

4.Package gui.page

4.1 Class Gameplay

4.1.1 Fields

Name	Description
- Stage primaryStage	Current Stage of the application.
- Canvas canvas	Base Canvas.
- GraphicsContext gc	Base GraphicsContext
- Scene scene	Base Scene

4.1.2 Constructor

Name	Description
+ Gameplay(Stage primaryStage)	Initialize primaryStage. Call setStage() Start a new game. (Using GameLogic) Show primaryStage.

4.1.3 Methods

Name	Description
+ void setStage()	Initialize canvas with dimension of 1280 by 720. Setup GraphicsContext. Initialize Scene. Set scene to current stage.
+ StackPane getGameplayPane()	Return this.
+ Stage getPrimaryStage()	Return primaryStage.

4.2 Class MainMenuPage

4.2.1 Constructor

Name	Description
+ MainMenuPage(Stage primaryStage)	Initialize mainMenuElement. Create new Scene. Set primaryStage scene to the created scene.

5. Package levels

5.1 Class LevelManager

5.1.1 Fields

Name	Description
- Image levelSprite	Store sprite that contains every type of tile in the level.
- int[][] levelData	Store tile data. The value of Red of every pixel in dataImage refers to the index of tile in levelSprite.

5.1.2 Constructor

Name	Description
+ LevelManager()	Initialize all fields. Call initializeLevel().

5.1.3 Methods

Name	Description

+ void initializeLevel()	Create new variable dataImage to store level data from current level. Use pixel reader to get R value from every pixel and store them in levelData.
+ void draw(GraphicsContext gc)	Draw Background. Call MapManager.drawElement Loop through levelData and draw each tile corresponding to the levelData at a specific location on the canvas.
+ int[][] getLevelData()	Return levelData.

5.2 Class MapManager

5.2.1 Methods

Name	**Description**
+ static void drawElement(int map, GraphicsContext gc)	Draw map's element: map = 0 -> draw Map1 map = 1 -> draw Map2 map = 2 -> draw Map3
+ static void drawElementMap1(GraphicsContext gc)	Draw map1
+ static void drawElementMap1First(GraphicsContext gc)	Draw map1's element first time
+ static void drawElementMap1Second(GraphicsContext gc)	Draw Map1's element second time
+ static void drawElementMap1Third(GraphicsContext gc)	Draw Map1's element third time

+ static void drawElementMap2(GraphicsContext gc)	Draw Map2's element
+ static void drawElementMap2First(GraphicsContext gc)	Draw Map2's element first time
+ static void drawElementMap2Second(GraphicsContext gc)	Draw Map2's element second time
+ static void drawElementMap2Third(GraphicsContext gc)	Draw Map2's element third time
+ static void drawElementMap3(GraphicsContext gc)	Draw Map3's element
+ static void drawElementMap3First(GraphicsContext gc)	Draw Map3's element first time
+ static void drawElementMap3Second(GraphicsContext gc)	Draw Map3's element second time
+ static void drawElementMap3Third(GraphicsContext gc)	Draw Map3's element third time
+ static void drawElementMap3Fourth(GraphicsContext gc)	Draw Map3's element fourth time
+ static void drawElementMap3Fifth(GraphicsContext gc)	Draw Map3's element fifth time

6.Package logic

6.1 Class GameLogic

6.1.1 Fields

Name	Description
- static GameLogic instance	store instance of GameLogic
- int currentLevel	Current level that the player is in.
- int score	Current score that the player has.
- Player player	Current Player.
- Playing playing	Current GameState playing
- Gameplay gameplay	Current Gameplay
- Scene scene	Current Scene
- AnimationTimer gameLoop	gameLoop that updates every entity.

6.1.2 Methods

Name	Description
+ void newGame(GraphicsContext gc, Scene scene, Gameplay gameplay)	Initialize all fields. Set the current score to 0. Set the current level to 1. If the gameLoop already exists, stop it. Start a new AnimationTimer. Set the current GameState to PLAYING.
+ void changeLevel()	Increment the level by 1. Initialize new Playing with the new level.

# void clearScreen(GraphicsContext gc)	Clear the screen of the GraphicsContext.

7.Package main

7.1 Class Main

7.1.1 Methods

Name	Description
+ void start(Stage primaryStage) throws Exception	Initialize MainMenuPage Set stage, and show
+ static void main(String[] args)	Main application

8.Package objects

8.1 Class ObjectManager

8.1.1 Fields

Name	Description
- ArrayList<PineappleAmmo> objects	List of objects currently active in the game.

8.1.2 Constructor

Name	Description
+ ObjectManager()	Initializes objects ArrayList.

8.1.3 Methods

Name	Description

+ void update(GraphicContext gc)	Update every object in objects ArrayList and remove inactive objects from the ArrayList.
+ void draw(GraphicsContest gc)	Draw every object in ArrayList.
+ ArrayList<PineappleAmmo> getCurrentAmmo()	return objects ArrayList.

8.2 Class PineappleAmmo

8.2.1 Fields

Name	Description
- boolean activated	Is this bullet activated?
- Player player	Active Player in the current game.
- int direction	Direction of the bullet -1 : left 1 : right

8.2.2 Constructor

PineappleAmmo(int x, int y, boolean activate, Player player, int direction)	Initialize all fields.

8.2.3 Methods

Name	Description
+ void draw(GraphicsContext gc)	Draw pineappleAmmo image at coordinate (x,y).
+ void update(GraphicsContext gc)	Update the hitbox and position of the bullet.

	If the bullet hits the player, do damage to the player and set activated to false.
+ void updatePosition()	Update the X Position of the bullet. If the bullet hits the wall or goes over the screen edge, set activated to false.
+ boolean isActivated()	Return activated value.

9.Package sharedObject

9.1 Interface IRenderable

9.1.1 Methods

Name	**Description**
+ void draw(GraphicsContext gc)	Draw anything

9.2 Class RenderableHolder

This class contain all resources such as Image, Font, Audio, Map

9.2.1 Methods

Name	**Description**
+ static void loadResource()	Load all resources

10.Package utils

10.1 Class Animations

10.1.1 Methods

Name	**Description**
+ static ArrayList<Image> getMushroomIdleLeft()	Return mushroom idle left animations
+ static ArrayList<Image>	Return mushroom idle left animations

getMushroomIdleRight()	
+ static ArrayList<Image> getMushroomAttackLeft()	Return mushroom attack left animations
+ static ArrayList<Image> getMushroomAttackRight()	Return mushroom attack left animations
+ static ArrayList<Image> getMushroomJumpLeft()	Return mushroom Jump left animations
+ static ArrayList<Image> getMushroomJumpRight()	Return mushroom Jump right animations
+ static ArrayList<Image> getMushroomWalkLeft()	Return mushroom walk left animations
+ static ArrayList<Image> getMushroomWalkRight()	Return mushroom walk right animations
+ static ArrayList<Image> getTomatoIdle()	Return tomato idle animations
+ static ArrayList<Image> getTomatoWalk()	Return tomato walk animations
+ static ArrayList<Image> getTomatoHit()	Return tomato hit animations
+ static ArrayList<Image> getPineappleIdle()	Return pineapple idle animations
+ static ArrayList<Image> getPineappleAttack()	Return pineapple attack animations
+ static ArrayList<Image> getPineappleHit()	Return pineapple hit animations

10.2 Class Constants

10.2.1 static class UniversalConstants

10.2.1.1 Fields

Name	Description
+ static final int TILE_SIZE	Tile size
+ static final int X_DIMENSION	Max x coordinate
+ static final int Y_DIMENSION	Max y coordinate

10.2.2 static class PlayerConstants

10.2.2.1 Fields

Name	Description
+ static final int IDLE_LEFT	Player's action animations
+ static final int IDLE_RIGHT	Player's action animations
+ static final int ATTACK_LEFT	Player's action animations
+ static final int ATTACK_RIGHT	Player's action animations
+ static final int JUMP_LEFT	Player's action animations
+ static final int JUMP_RIGHT	Player's action animations
+ static final int WALK_LEFT	Player's action animations
+ static final int WALK_RIGHT	Player's action animations

+ static final int DEATH	Player’s state
+ static final int PLAYER_ATTACK_RANGE	Player’s attack range
+ static final int X_DRAW_OFFSET	Offset X coordination
+ static final int Y_DRAW_OFFSET	Offset Y coordination
+ static final int HITBOXWIDTH	Hitbox width
+ static final int HITBOXHEIGHT	Hitbox height
+ static final float AIRSPEED	Air speed
+ static final float GRAVITY	Gravity
+ static final float JUMP_SPEED	Player’s Jump speed
+ static final float FALL_SPEED_AFTER_COLLISION	Fall speed
+ static final float KNOCKBACK_SPEED	Knockback speed
+ static final int DAMAGE	Player’s damage
+ static final int HEALTH	Player’s health

10.2.2.2 Methods

Name	**Description**
+ static int GetSpriteAmount(int playerAction)	Return amount for each action animations

10.2.3 static class Directions

10.2.3.1 Fields

Name	Description
+ static final int RIGHT	Player’s direction
+ static final int LEFT	Player’s direction

10.2.4 static class EnemyConstants

10.2.4.1 Fields

Name	Description
+ static final int TOMATO	Enemy’s type
+ static final int PINEAPPLE	Enemy’s type
+ static final int ATTACK_RANGE	Enemy’s attack range
+ static final int ATTACK_DAMAGE	Enemy’s attack damage
+static final int X_OFFSET	Offset X coordinate
+ static final int Y_OFFSET	Offset Ycoordinate
+ static final int TOMATO_X_OFFSET	Tomato’s offset X coordinate
+ static final int TOMATO_Y_OFFSET	Tomato’s offset Y coordinate
+ static final int PINEAPPLE_X_OFFSET	Pineapple’s offset X coordinate
+ static final int PINEAPPLE_Y_OFFSET	Pineapple’s offset Y coordinate
+ static final int TOMATO_HITBOX_WIDTH	Tomato’s hitbox width
+ static final int TOMATO_HITBOX_HEIGHT	Tomato’s hitbox height

+ static final int PINEAPPLE_HITBOX_WIDTH	Pineapple hitbox width
+ static final int PINEAPPLE_HITBOX_HEIGHT	Pineapple hitbox height
+ static final int TOMATO_R_VALUE	Tomato's red value of tile
+ static final int PINEAPPLE_R_VALUE	Tomato's red value of tile
+ static final int TOMATO_HEALTH	Tomato's health
+ static final int PINEAPPLE_HEALTH	Pineapple's health
+ static final int IDLE	Enemy's action
+ static final int ATTACK	Enemy's action
+ static final int WALK	Enemy's action
+ static final int HIT	Enemy's action

10.2.4.2 Methods

Name	**Description**
+ static int GetSpriteAmount(int Type, int State)	Return amount of each action animations (each type)

10.3 Class HelperMethods

10.3.1 Methods

Name	**Description**
+ static boolean CanMoveHere(float x, float y, int width, int height, int[][]	Check whether every corner of the hitbox rectangle at coordinate (x,y) is not solid. Return

	levelData)	true if every corner of the hitbox rectangle is not solid. Return false otherwise.
-	static boolean IsSolid(float x, float y, int[][] levelData)	Check whether the Tile at coordinate (x,y) is Solid.
+	static boolean InClearZone(float x, float y, int width, int height, int[][] levelData)	Check whether the bottom right corner of the hitbox rectangle at coordinate (x,y) is in level clear zone.
-	static boolean ClearLevelZone(float x, float y, int[][] levelData)	Check whether the Tile at coordinate (x,y) is in the proceed-to-next-level zone.
+	static float GetEntityXPosNextToWall(Rectangle hitbox, float xSpeed)	Calculate the X position of the player if their hitbox hit the wall.
+	static float GetEntityPosRoofFloor(Rectangle hitbox, float airSpeed)	Calculate the Y position of the player if their hitbox hit the ceiling or the floor.
+	static boolean isEntityOnFloor(Rectangle hitbox, int[][] levelData)	Check whether the 2 bottom corners of the hitbox rectangle are on the floor. Return true if they do. Return false otherwise.