Witch’s Garden

document

Created by

Naphat Serirak 6632061321

Raksakul Hiranas 6632190621

2110215 Programming Methodology
Semester 2 Year 2023
Chulalongkorn University

Witch’s garden

Introduction

"Witch’s Garden" is an action-packed RPG game where you play as a witch collecting veggies to make potions to reclaim her lost magic. You can use the broom and manipulate the weather to protect the veggies from slimes.

Rules

There are 3 levels you need to complete to win this game. Each level represents the ingredients for one potion.

- Level 1 : LOVE POTION
- Level 2 : STAR POTION
- Level 3 : LUNA POTION

In each level, you must collect veggies to create potions. Each level has a different time and list of veggies to collect. To complete the game , you must collect all the listed veggies before time is running out.

Veggie has **Hp, water point, and growth point**. You can collect the veggie if that veggie has grown enough. There are slimes in the game that attack veggies which will decrease veggie’s Hp. The water point of veggies decreases time by time. If the Hp or water point of a veggie is less than or equal to 0 , that veggie will die.

Types of veggies in this game :

Daffodil : drops water very fast

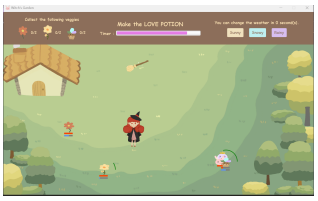
RedFlower : grows very fast

RainbowDrake : grows very slow

There is a **weathering system** in this game which has **SUNNY, SNOWY, and RAINY**. Each weather has a different stats for most elements in this game. For example, a player will walk slowly but hit very hard in SNOWY. But, the most important part of this system is because Veggie has the water point which decreases every single time. You have to change the weather to RAINY to maximize the water point of every veggie, otherwise that veggie may die of losing hydration. But you can't just stay in Rainy weather because it will gradually create white vision which makes it very hard to play.

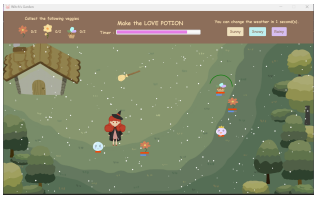
The stats of each weather :

SUNNY



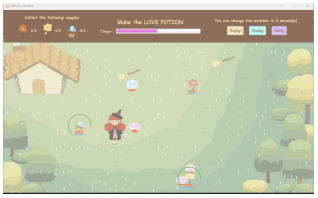
- The witch : walk normal , hit slime very fast but not strongly
- Slime : walk normal , attack veggie normal
- Veggies : grows normal , water drops very fast

SNOWY



- The witch : walk slowly , hit strongly but not fast
- Slime : walk slowly , attack veggie strongest
- Veggies : grows slowly , water drops slowly

RAINY



- The witch : walk fastest , balance hit
- Slime : walk fastest , attack veggie weakest
- Veggies : grows fast , water will not drop
- Maximize water points of veggies
- Create gradually appear white screen

Slime is randomly spawn on the map. The maximum number of slimes is based on the level. The higher the level, the higher the number of slimes, the more difficult.

There are 3 types of slimes in the game

Normal Slime :
walk normal , hit normal , normal Hp

Fast Slime :
walk fast , hit normal , normal Hp

HitHard Slime :
walk normal , hit strongly , high Hp

There are the brooms randomly spawn on the map. You have to get Broom first before attacking slime. Each broom has random durability which will be decreased per attack of slime. So, you need to get the new broom on the map , if it has broken.

Example / How To Play

WASD to walk

Change the weather in the map by **LEFT CLICK** on the weather button. Wait for the time to be ready for changing weather first.

Press **E** to collect veggies.

- The green circle represents the growth state. You can collect veggies if the green circle around veggies disappears.

press **E** to get Broom

SPACEBAR to use Broom to attack slime

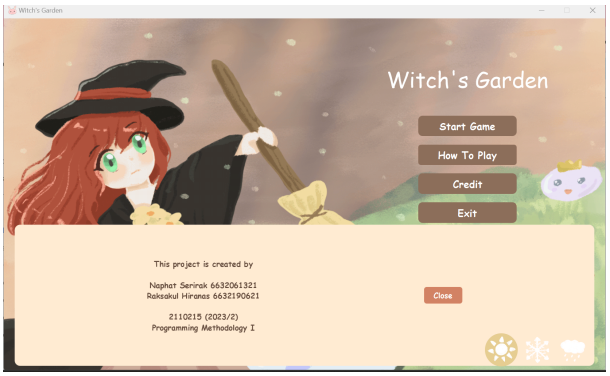
Slimes will come to attack veggies. Make sure to use your broom to protect them. Use the weather manipulation power to help you save them.

Scene

MainMenu Scene

This is the main menu scene. You can start the game, see how to play, see credit and exit the game.

default main menu scene



How To Play and Credit

You can change the theme of the main menu scene by clicking the weather button at the bottom right of the screen.

SUNNY theme

SNOWY theme

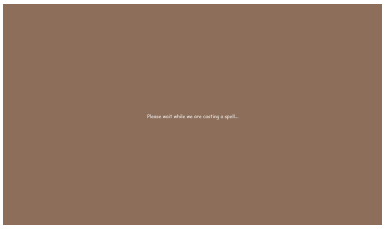
RAINY theme

PreStory scene

This is the story before starting the game which is about the witch losing her magic , except her weather manipulation power, and about making the potions for regaining her power back.

Some of the scenes in the PreStory scene.

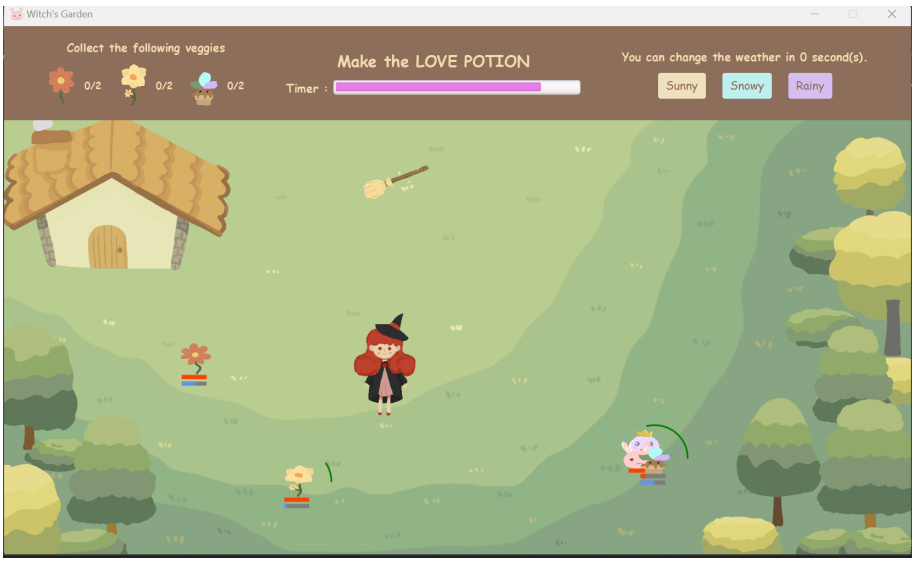
Loading Scene



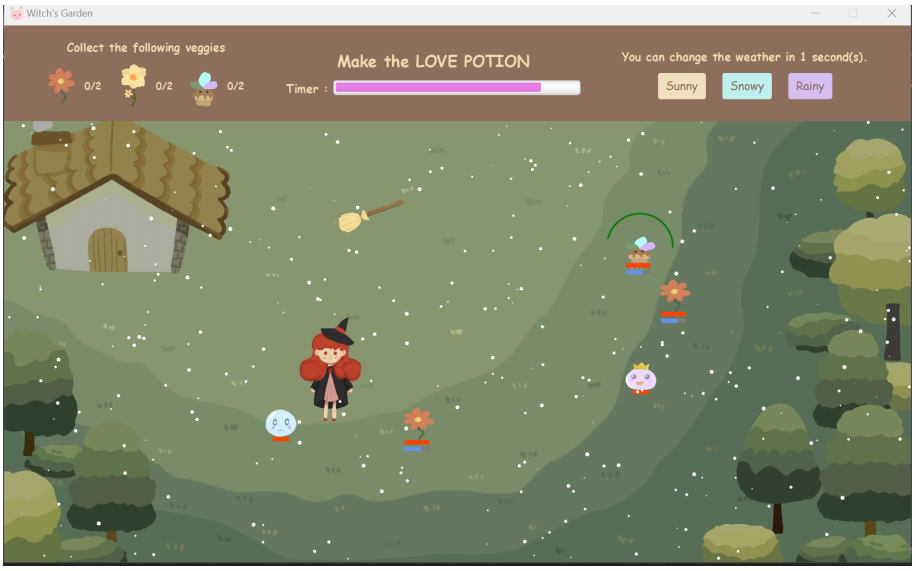
This is a loading scene which appears before starting the game to load resources.

Game Scene

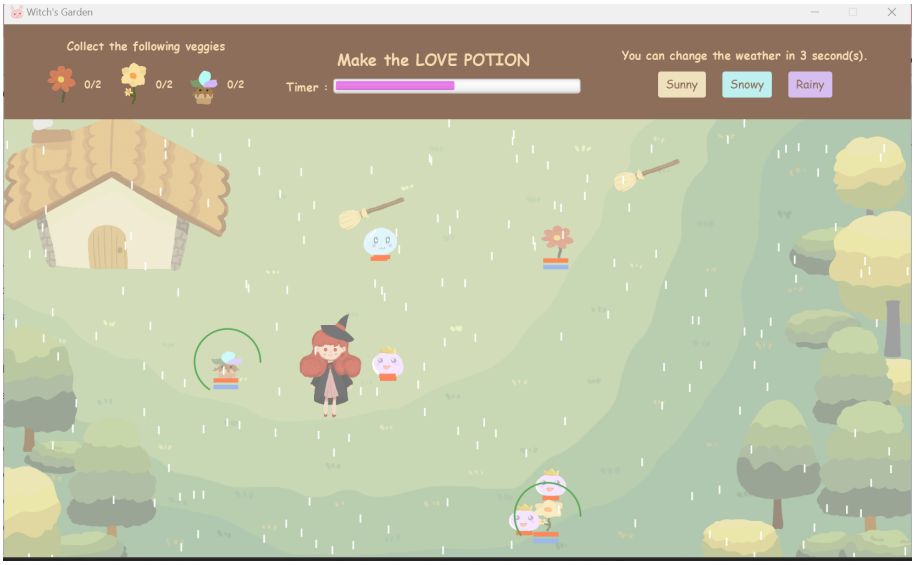
This is a game scene where you can collect veggies, attack slime , and change the weather here.



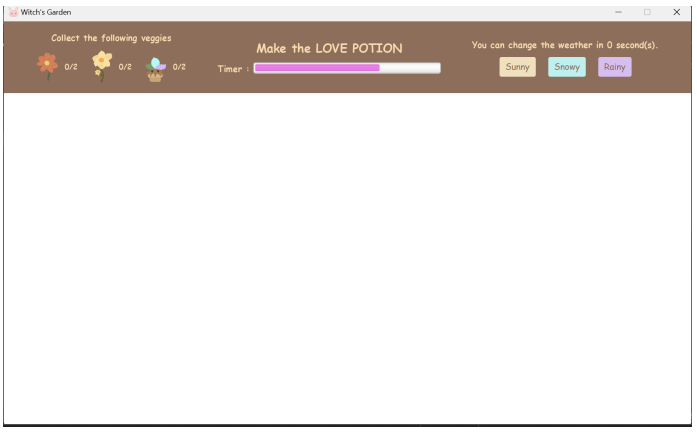
game scene in SUNNY



game scene in SNOWY



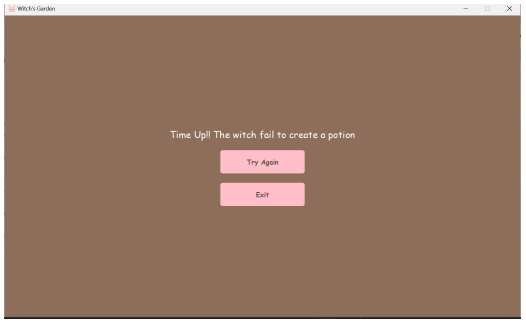
game scene in RAINY

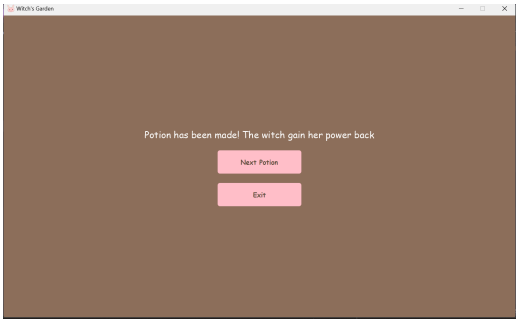


In RAINY , the game screen will gradually appear white vision until the map cannot be seen.

LevelEnd Scene

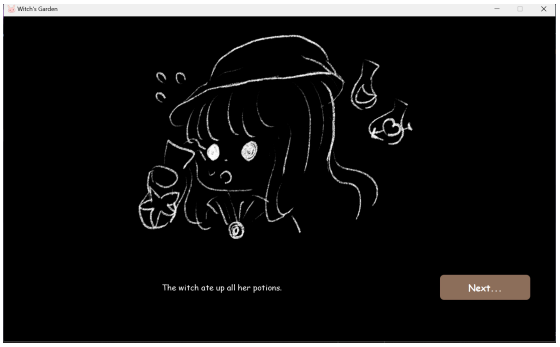
This is the scene that appears at the end of each level. The message depends on if the game is winning or not.

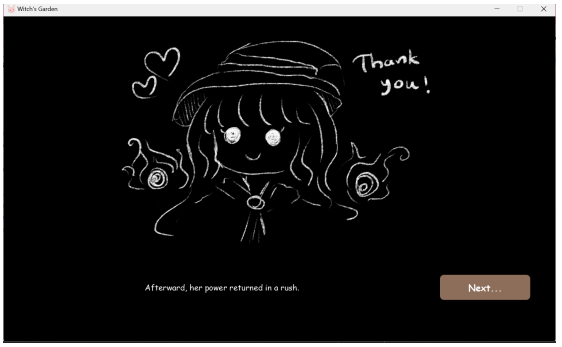




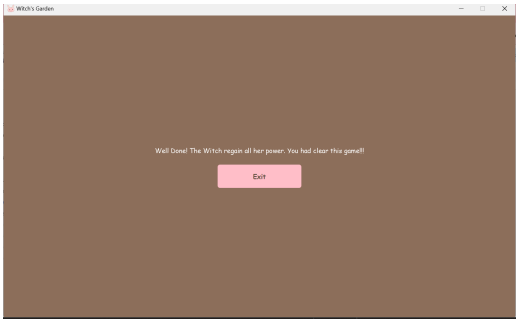
EndStory Scene

This scene is shown after you have completed the game. It is the story about the witch getting back her power after drinking potions.





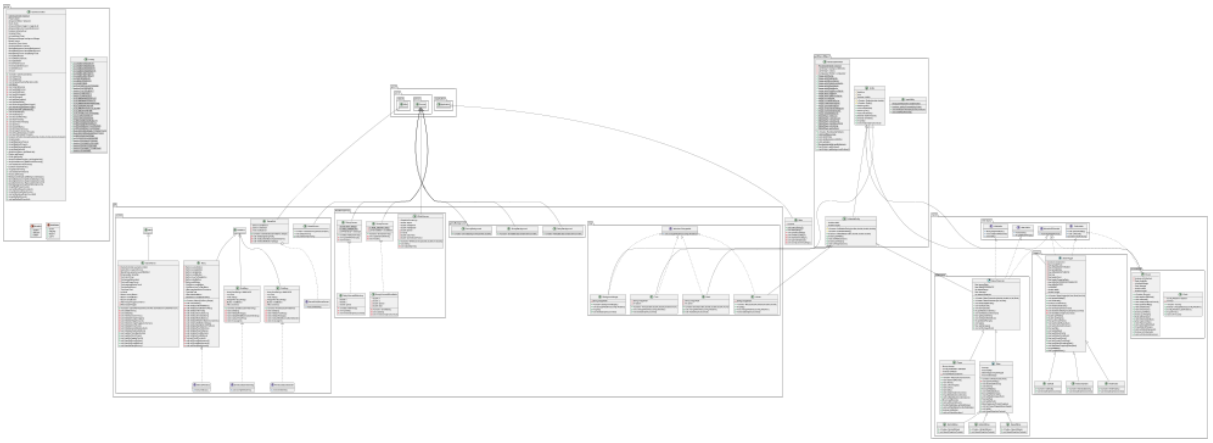
GameEnd Scene



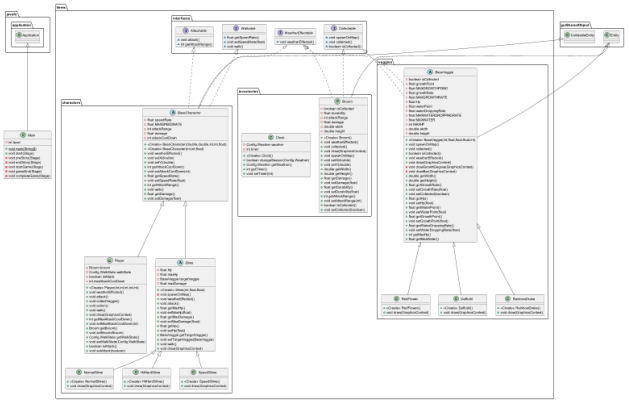
After completing the game and story, this scene will be shown for you to exit the game.

Class Diagram

(full HD photo in https://github.com/2110215-ProgMeth/project-cp-2023-2-indexoutofbound)

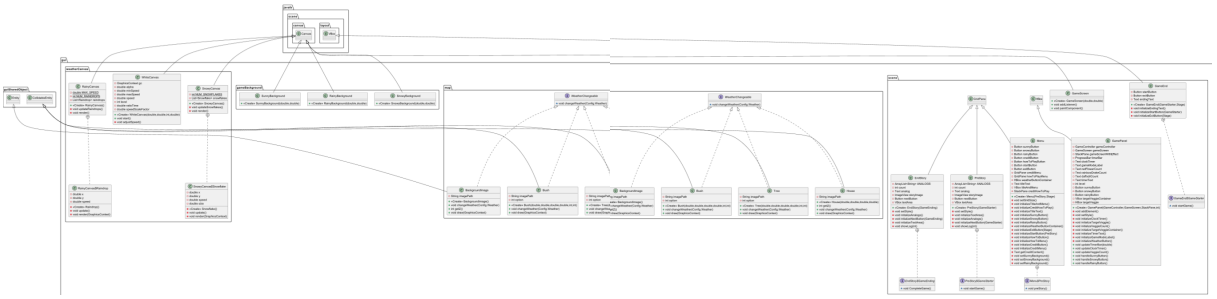


Class diagram of Items and main

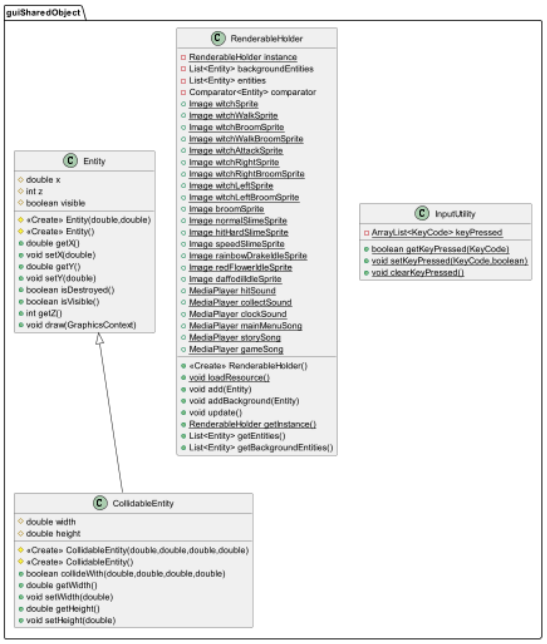


Class diagram of Game

Class diagram of GUI



Class diagram of GUISharedObject



1. Package items

1.1. Package characters

1.1.1 public abstract class BaseCharacter extends CollidableEntity implements Walkable, Attackable, WeatherEffectable

Fields

- float speedRate	Walking speed rate
- final float MAXSPEEDRATE	Max walking speed rate
- int attackRange	Attack range of character
- float damage	Damage that character made at that time
- int attackCooldown	Cooldown timer for attack action

Constructor

+ BaseCharacter(double positionX, double positionY, int maxSpeedRate, int attackRange, float damage)	- receive positionX, positionY, maxSpeedRate, attackRange,Damage - construct a CollidableEntity with super(positionX, positionY, 0,0) - set MAXSPEEDRATEm attackRange, damage
+ BaseCharacter(int maxSpeedRate, int attackRange, float damage)	- receive maxSpeedRate, attackRange,Damage - construct a CollidableEntity with super() - set MAXSPEEDRATEm attackRange, damage

Method

+ void setX(double X)	set positionX to Math.max(0,Math.min(x, Config.GAMEFRAMEWIDTH))
+ void setY(double Y)	set positionY to Math.max(0,Math.min(y, Config.GAMEFRAMEHEIGHT))
+ void weatherEffected()	- get weatherNow from GameController - set speedRate base on weather
+ int getAttackCooldown()	Returns the attack cooldown value
+ void setAttackCooldown(int attackCooldown)	Sets the attack cooldown, ensuring it is non-negative.
+ float getSpeedRate()	Returns the speed rate

+ void setSpeedRate(float speedRate)	Sets the speed rate within the range of 0 to MAXSPEEDRATE.
+ int getAttackRange()	Returns the attack range.
+ void setAttackRange(int attackRange)	Sets the attack range, ensuring it is non-negative.
+ float getDamage()	Returns the damage value.
+ void setDamage(float damage)	Sets the damage value, ensuring it is non-negative.

1.1.2 public abstract class Slime extends BaseCharacter

Fields

- float HP	Slime's HP
- float maxHp	Slime's max HP
- private BaseVeggies targetVeggie	Slime will walk to and attack target veggie
- float maxDamage	Slime's max damage

Constructor

+ Slime(int speedRate, float maxHp, float maxDamage)	- receive maxSpeedRate, maxHp, maxDamage - construct a BaseCharacter with speedRate, random maxSpeed, maxDamage - set maxHp to maxHP - set HP tp maxHP - set targetVeggie from one of veggiesList in GameController - set width to Config.SLIMEWIDTH - set height to Config.SLIMEHEIGHT - calls spawnOnMap() - set Z position to 400

Method

+ void attack()	- If attack cooldown > 0, slime won't attack - If target veggie not in GameController's List, slime will find new target - if distance to < attackRange, slime will attack

+ void weatherEffected()	- get weatherNow from GameController if weather is SUNNY, set speedRate = 0.6 * MAXSPEEDRATE if weather is RAINY, set speedRate = 0.7 * MAXSPEEDRATE if weather is SNOWY, set speedRate = 0.5 * MAXSPEEDRATE
+ void walk()	- Calculate distances (disX and disY) between the entity and its target. - Compute the total distance by euclidean distance - if distance - this.getAttackRange() > Config.SLIMEWALKSTEP - calculate new positions based on current position, speed rate, and direction. - verify accessibility of new position. - update entity's position if accessible.
+ void draw(GraphicsContext gc)	- use slime image base on its type if it's NormalSlime, use normalSlimeSprite if it's HitHardSlime, use hitHardSlimeSprite if it's SpeedSlime, use speedSlimeSprite - draw slime image in position X = getX() - getWidth()/2 and position Y = getY() - getHeight()/2 - draw HPbar. set HPbar's width to 20 px. - HPPercentage = getHp() / getMaxHp() - set HPbar color to Color.ORANGERED, and background to Color.GRAY
- void spawnOnMap()	random position X and Y in map within the spawn bound. This will be randomed until it finds the proper position which does not collide with elements in the map.
+ float getMaxHp()	Returns the maximum HP value
+ void setMaxHp(float maxHp)	Sets the maximum HP value
+ float getMaxDamage()	Returns the maximum damage value
+ void setMaxDamage(float maxDamage)	Sets the maximum damage value
+ float getHp()	Returns the current HP value
+ void setHp(float hp)	Sets the current HP value, ensuring it remains within the range of 0 to maxHp.
+ BaseVeggies getTargetVeggie()	Returns the target vegetable entity
+ void setTargetVeggie(BaseVeggies targetVeggie)	Sets the target vegetable entity

1.1.3 public class NormalSlime extends Slime

constructor

+ NormalSlime()	- use super constructor in Slime - set speed to Config.SLIMEMINSPEEDRATE - set HP to 5 - set maxDamage to 2

method

+ void draw(GraphicsContext gc)	- use super draw() in Slime - draw Image using normalSlimeSprite

1.1.4 public class HitHardSlime extends Slime

constructor

+ NormalSlime()	- use super constructor in Slime - set speed to Config.SLIMEMINSPEEDRATE - set HP to 12 - set maxDamage to 5

method

+ void draw(GraphicsContext gc)	- use super draw() in Slime - draw Image using hitHardSlimeSprite

1.1.5 public class SpeedSlime extends Slime

constructor

+ NormalSlime()	- use super constructor in Slime - set speed to Config.SLIMEMINSPEEDRATE +10 - set HP to 5 - set maxDamage to 1

method

+ void draw(GraphicsContext gc)	- use super draw() in Slime - draw Image using speedSlimeSprite

1.1.6 public class Player extends BaseCharacter

field

- Broom broom	witch’s current broom
- Config.WalkState walkState	witch’s walk state which are STAY, FRONT, RIGHT, LEFT
- boolean isAttack	which’s attack state
- int maxAttackCoolDown;	witch's max cooldown time after attack

method

+ Player(int positionX, int positionY, int maxSpeedRate, int attackRange, int damage)	- Initializes a Player object by super constructor - Sets the position (positionX, positionY), maximum speed rate (maxSpeedRate), attack range (attackRange), and damage (damage) of the player. - Initializes the player's broom to null. - Sets the walk state of the player to STAY. - Sets the width and height of the player based on predefined constants. - Sets the z-coordinate of the player to 999.
+ void weatherEffected()	- get weather from the gameController - If the weather is SUNNY, set maximum attack cooldown to 0.5 * PLAYERCOOLDOWNTIME - If the weather is RAINY, set maximum attack cooldown to 0.7 * PLAYERCOOLDOWNTIME - If the weather is SNOWY, set maximum attack cooldown to 1* PLAYERCOOLDOWNTIME
+ void attack()	- checks conditions for attacking: - If the attack cooldown is greater than 0, - If the SPACE key is pressed, - If the player does have a broom. - attacks the slime within the broom's attack range, make a threads that will - Plays a hit sound - reduces the slime's HP when hit. - Reduces the broom's durability - player's attack cooldown to maxAttackCoolDown

	- setAttack true - sleep for 300 millisecond - setAttack false
+ void collectVeggie()	- allows the player to collect veggies when the E key is pressed. - checks if the player is within range of any veggies. - removes the collected veggies from the game controller's veggies list.
+ void walk()	- overrides the walk method from the superclass. - check WASD keys to set a walk direction - adjusts movement speed for diagonal walks - checks for collisions with game elements before moving the player.
+ void draw(GraphicsContext gc)	- sets the appropriate sprite animation based on the player's actions and equipment. - draws the remaining attack cooldown if the player has a broom equipped.
+ void action()	- call weatherEffects() - call walk() - call attack() - call collectVeggie()
+ getMaxAttackCoolDown()	return player’s maximum attack cooldown
+ setMaxAttackCoolDown(int maxAttackCoolDown)	set player’s maximum attack cooldown
+ getBroom()	return player’s broom
+ setBroom(Broom broom)	set player’s broom
+ getWalkState()	return player’s walk state
+ setWalkState(Config.WalkState walkState)	set player’s walk state
+ isAttack()	return attack state
+ setAttack(boolean attack)	set attack state

1.2. Package interfaces

1.2.1. public interface Attackable

method

void attack()	abstract method for attack
int getAttackRange()	abstract method for get attack range

1.2.2. public interface Collectable

method

void spawnOnMap()	abstract method for spawn item on map
void collected()	abstract method for collect item
boolean isCollected()	abstract method to check if item is collected

1.2.1. public interface Walkable

method

float getSpeedRate()	abstract method for get speed rate
void setSpeedRate(float speedRate)	abstract method for set speed rate
void walk()	abstract method for walk

1.2.1. public interface WeatherEffectable

method

void weatherEffected()	abstract method for weather effected

1.3. Package Inventories

1.3.1. public class Broom extends Entity implements Collectable, WeatherEffectable

field

- boolean isCollected	broom’s collected state
- float durability	broom’s durability
- int attackRange	broom’s attack range
- float damage	broom’s damage
- double width	broom’s width

- double height	broom’s height

constructor

+ Broom()	- Initializes a broom object with random durability, attack range, and damage. - Sets the width and height of the broom based on CONFIG constants. - Spawns the broom on the map. - Sets the z-coordinate of the broom to 300.

method

+ void weatherEffected()	- Updates the broom's damage based on the current weather condition - If the weather is SUNNY, set maximum attack cooldown to 0.5 * BROOMDAMAGEPERATTACK - If the weather is RAINY, set maximum attack cooldown to 0.75 * BROOMDAMAGEPERATTACK - If the weather is SNOWY, set maximum attack cooldown to 1* BROOMDAMAGEPERATTACK
+ void collected()	- Checks if the player is attempting to collect the broom. - Checks if the player is within range to collect the broom. - Assigns the broom to the player if the player does not already have a broom. - Plays a collect sound effect and marks the broom as collected. - Removes the broom from the game Scene
+ void draw(GraphicsContext gc)	- Draw the broom sprite on the canvas at the current position.
+ void spawnOnMap()	- Spawns the broom at a random position within the game boundaries. - Ensures the broom spawns in an accessible position on the map.
+ void setX(double x)	- Sets the x-coordinate of the broom while ensuring it stays within the game boundaries
+ void setY(double y)	- Sets the y-coordinate of the broom while ensuring it stays within the game boundaries.

+ double getWidth()	return broom’s width
+ double getHeight()	return broom’s height
+ float getDamage()	broom's damage
+ void setDamage(float damage)	sets the broom's damage
+ float getDurability()	Returns the broom's durability
+ void setDurability(float durability)	Sets the broom's durability
+ int getAttackRange()	Returns the broom's attack range
+ void setAttackRange(int attackRange)	Sets the broom's attack range
+ boolean isCollected()	Checks if the broom has been collected by the player
+ void setCollected(boolean collected)	Sets the collected state of the broom

1.3.2. public class Clock

field

- Config.Weather weather	current weather of clock
- int timer	weather cooldown timer

constructor

+ Clock()	- set timer to CLOCKCOOLDOWNTIME - set weather to sunny

method

+ boolean changeSeason(Config.Weather weather)	- Checks if the weather can be changed - Changes weather to sending param
+ Config.Weather getWeather()	Returns the current weather
+ int getTimer()	Returns the current cooldown timer
+ void setTimer(int timer)	Sets the cooldown timer for the clock

1.4. Package veggies

1.4.1. public abstract class BaseVeggie extends Entity implements WeatherEffectable, Collectable

field

- boolean isCollected	veggie’s collect state
- float growthPoint	veggie’s growth point
- final float MAXGROWTHPOINT	veggie’s max growth point
- float growthRate	veggie’s growth rate
- final float MAXGROWTHRATE	veggie’s max growth rate
- float Hp	veggie’s hp
- float waterPoint	veggie’s water point
- float waterDroppingRate	veggie’s water dropping rate
- final float MAXWATERDROPPINGRATE	veggie’s max dropping rate
- final float MAXWATER	veggie’s max water
- final int MAXHP	veggie’s max hp
- double width	veggie’s width
- double height	veggie’s height

constructor

+ BaseVeggie(int hp, float maxWater, float growthRate, float waterDroppingRate, int maxGrowthPoint)	- set MAXHP - set HP to MAXHP - set MAXWATER - set waterPoint to MAXWATER - set MAXGROWTHRATE - growthRate to MAXGROWTHRATE - set MAXWATERDROPPINGRATE - set waterDroppingRate to MAXWATERDROPPINGRATE - set growthPoint to 0 - set MAXGROWTHPOINT - Spawns vegetable on map. - Sets z-coordinate to 600. - Sets width and height based on CONFIG

method

+ void spawnOnMap()	Spawns the vegetable on a random accessible position on the map.

+ void collected()	- Checks if the vegetable has reached its maximum growth point - if true, collect it by the game controller.
+ boolean isCollected()	Check if the vegetable has been collected.
+ void weatherEffected()	Adjusts the growth rate and water dropping rate of the vegetable based on the current weather.
+ void draw(GraphicsContext gc)	Draws the vegetable's growth degree and growth bar
+ void drawGrowthDegree(GraphicsContext gc)	Draws an arc representing the vegetable's growth degree.
+ void drawBar(GraphicsContext gc)	Draws the health and water bars for the vegetable
+ double getWidth()	Get veggie’s width
+ double getHeight()	Get veggie’s height
+ float getGrowthRate()	Get veggie’s growth rate
+ void setGrowthRate(float growthRate)	Set veggie’s growth rate
+ void setCollected(boolean collected)	Get veggie’s collected
+ float getHp()	Get veggie’s HP
+ void setHp(float hp)	Set veggie’s HP
+ float getWaterPoint()	Get veggie’s water point
+ void setWaterPoint(float waterPoint)	Set veggie’s water point
+ float getGrowthPoint()	Get veggie’s growth point
+ void setGrowthPoint(float growthPoint)	Set veggie’s growth point
+ float getWaterDroppingRate()	Get veggie’s water dropping rate
+ void setWaterDroppingRate(float waterDroppingRate)	Set veggie’s water dropping rate
+ int getMaxHp()	Get veggie’s max HP
+ float getMaxWater()	Set veggie’s max HP

1.4.2. public class Daffodil extends BaseVeggie

constructor

+ Daffodil()	- call super constructor - set hp = 20 - set maxWater = 200 - set growthRate = 6 - set waterDroppingRate = 20 - set maxGrowthPoint = 33

method

+ void draw(GraphicsContext gc)	- use super draw() in BaseVeggie - draw Image using daffodilSprite

1.4.3. public class RainbowDrake extends BaseVeggie

constructor

+ RainbowDrake()	- call super constructor - set hp = 15 - set maxWater = 200 - set growthRate = 6 - set waterDroppingRate = 10 - set maxGrowthPoint = 42

method

+ void draw(GraphicsContext gc)	- use super draw() in BaseVeggie - draw Image using rainbowDrakeSprite

1.4.4. public class RedFlower extends BaseVeggie

constructor

+ RedFlower()	- call super constructor - set hp = 15 - set maxWater = 200 - set growthRate = 7 - set waterDroppingRate = 10 - set maxGrowthPoint = 18

method

+ void draw(GraphicsContext gc)	- use super draw() in BaseVeggie - draw Image using redFlowerSprite

2. Package game

2.1. public class Config

enum

+ enum Weather	SUNNY, SNOWY, RAINY
+ enum WalkState	STAY, FRONT, RIGHT, LEFT

field

(all UPPERCASE represent **final** variable, all underline represent **static** variable)

+ final int GAMEFRAMEWIDTH	game frame width is 1100
+ final int GAMEFRAMEHEIGHT	game frame height is 650
+ final int GAMESCREENWIDTH	game screen width is 1100
+ final int GAMESCREENHEIGHT	game screen height is 535
+ final int GAMELABELWIDTH	game label width is 1100
+ final int GAMELABELHEIGHT	game label heigh is 115
+ final int WIDTHPERROW	width per row is 20
+ final int HEIGHTPERROW	height per row is 20
+ final int GAMETIMER	default game timer is 120
+ final int PLAYERCOLLECTRANGE	player collect range is 70
+ final double PLAYERWIDTH	player width is 90
+ final double PLAYERHEIGHT	player height is 128.6
+ final double SLIMEWIDTH	slime width is 38
+ final double SLIMEHEIGHT	slime height is 34
+ final int SLIMEMAXSPEEDRATE	slime max speed rate is 20
+ final int SLIMEMINSPEEDRATE	slime min speed rate is 15
+ final int SLIMEMAXDAMAGERANGE	slime max damage range is 30
+ final float SLIMEWALKSTEP	slime walk step is 2
+ final int SLIMEATTACKCOOLDOWN	default slime attack cooldown is 1
+ final int SLIMESPAWNTIME	slime spawn time is 5
+ final int PLAYERCOOLDOWNTIME	player cooldown time is 100 ms

+ final double BROOMWIDTH	broom width is 90
+ final double BROOMHEIGHT	broom height is 45
+ final int BROOMMAXDURABILITY	broom max durability is 20
+ final int BROOMMINDURABILITY	broom min durability is 10
+ final int BROOMMAXATTACKRANGE	broom max attack range is 100
+ final int BROOMMINATTACKRANGE	broom min attack range is 60
+ final float BROOMDURABILITYPERATTACK	broom durabuluty per attack is 1.5
+ final float BROOMDAMAGEPERATTACK	broom damage per attack is 3
+ final int BROOMSPAWNTIME	broom spawn time is 15
+ final int CLOCKCOOLDOWNTIME	clock cool dowm time is 5
+ final double SPAWNLEFTBOUND	spawn left bound is 80
+ final double SPAWNRIGHTBOUND	spawn right bound is 980
+ final double SPAWNTOPBOUND	spawn top bound is 50
+ final double SPAWNBOTTOMBOUND	spaw bottom bound is GAMESCREENHEIGHT - 50
+ final double VEGGIESIZE	veggie size is 40

2.2. public class GameController

field

- GameController instance	Game Controller instance
- Player player	player that user can control
- ArrayList<Slime> slimeList	slime list in a game
- Clock clock	clock for weather change
- ArrayList<BaseVeggie> veggiesList	veggie list in a game
- ArrayList<Broom> broomOnGround	broom that spawn in game map
- boolean isGameOver	is game over boolean variable
- int gameTimer	game timer
- int maxGameTimer	maximum game timer

- BackgroundImage backgroundImage	background image
- House house	house element in game map
- ArrayList<Tree> trees	trees elements in game map
- ArrayList<Bush> bushes	bushes elements in game map
- SunnyBackground sunnyBackground	sunny background
- SnowyBackground snowyBackground	snowy background
- RainyBackground rainyBackground	rainy background
- int maxRedFlower	max Red Flower in each game
- int maxRainbowDrake	max Rainbow Drake in each game
- int maxDaffodil	max Daffodil in each game
- int redFlowerCount	collected Red Flower in each game
- int rainbowDrakeCount	collected Rainbow Drake in each game
- int daffodilCount	collected Daffodil in each game
- int level	game current level

constructor

+ GameController()	- add background image and house - calls addTree() , addBush() , initializeWeatherBackground()

method

+ <u>GameController getInstance()</u>	Create new GameController if there no instance return GameController instance
- void addTree()	add tree to trees , and iterate over trees and call RenderableHolder.getInstance().addBackground(tree)
- void addBush()	add bush to bushes , and iterate over bushes and call RenderableHolder.getInstance().addBackground(bush)
- void initializeWeatherbackground()	calls the constructor to sunnyBackground , snowyBackground ,and rainyBackground

+ void clearStats(in level)	clear game stats before starting new level - clear all keypress of input utility - Calls startThread()
- void deleteOldEntity()	Removes old entities
- setStats(int level)	- Initializes a new player - Resets the lists for vegetables, slimes, and brooms on the ground. - Creates a new Clock instance. - Sets `isGameOver` to false - Calculates the game timer and maximum counts based on the level - set counts for each type of veggie to 0 - set level value
+ void startThread()	calls startTimerThread() , startPlayerActionThread() , and startSlimeWalkThread()
- void startTimerThread()	- start new thread - if it's not gameOver yet calls Thread.sleep(1000) , spawn() , decrease timer by 1 , updateStats() , and checkGameOver()
- void spawn()	- spawn broom every Config.BROOMSPAWNTIME - spawn slime every Config.SLIMESPAWNTIME and limit slime in the map to not exceed 2 * level + 3
- void updateStats()	- update slime attack cooldown decreased by 1 - check if weather is RAINY , if it is , set water point to max ,else , decreased water point by water dropping rate - increased growthPoint of each veggie by growthRate
- void checkGameOver()	- check if user collect all listed veggie yet , if it is, set GameOver to true - check if time is running out , if it is , set GameOver to true
- void startPlayerActionThread()	- start new thread - calls Thread.sleep(20) , game.getPlayer().action() , game.getPlayer().setAttackCoolDown(game.getPlayer().getAttackCoolDown() - 20)
- void startSlimeWalkThread()	- start new thread - calls Thread.sleep(100) - iterates over slimeList , calls slime.walk()
+ public static void play() throws InterruptedException	- Check if broom's duration == 0, if true set player's broom to null, else use weatherEffected() - Check if any slime has HP < 0, if true remove

	those slimes, else use weatherEffected() - Check if any veggie has waterPoint <= 0 or HP <= 0, if true call deleteVeggie() on the veggie, else use weatherEffected() - Check if any broom on the ground is collected, call collected() on each broom.
- void collectBroom()	iterate over broomOnGround and calls broom.collected()
- void updateBroom()	check if the durability of broom is less than or equal to 0 , if it is set broom of player to null., if its not call getInstance.getPlayer().getBroom().weatherEffected()
- void updateSlime()	- check if slime in slimeList ‘s Hp less than or equal to 0, if it is , remove the slime. - slime.weatherEffected()
- void updateVeggie()	- check if veggie in veggies List ‘s Hp or water point less than or equal to 0, if it is , remove the - calls slime.weatherEffected()
+void initGames()	Spawn initial veggies based on the level by calling getNewVeggie() for level + 2 times.
+ void getNewVeggie()	- Check if the player has collected the maximum number of each veggie type. - Randomly choose the type of veggie to spawn, ensuring not to exceed the maximum count for each type. - Create and add the new veggie to the game, updating the veggies list and rendering it on the screen.
+ void getNewSlime()	- Randomly select a slime type by generating a random number between 0 and 2 - Add the new slime to the slime list and render it in the game.
+ void collectVeggie(BaseVeggie veggie)	- increase veggie count base on collected veggie - Call deleteVeggie() to remove the veggie from the game.
+ void deleteVeggie(BaseVeggie veggie)	- Remove the veggie from the list of entities. Remove the veggie from the veggies list. - Call getNewVeggie() to spawn a new veggie.
+ boolean isPositionAccesible (double x, double y, double width, double height, boolean isPlayer)	- Check if the given position collides with any tree, bush, and house, if so, return false - If the entity is not a player, check if it collides with the player, if so, return false

	- return true.
+ int getLevel()	get game level
+ int getMaxGameTimer()	get max game timer
+ int getMaxRedFlower()	get max red flower
+ int getMaxRainbowDrake()	get max rainbow drake
+ int getMaxDaffodil()	get max daffodil
+ ArrayList<Slime> getSlimeList()	get game slime list
+ Player getPlayer()	get game player
+ Clock getClock()	get game clock
+ ArrayList<BaseVeggie> getVeggiesList()	get game veggie list
+ ArrayList<Broom> getBroomOnGround()	get game broom on ground
+ void setGameover(boolean isGameOver)	set game over value
+ boolean isGameOver()	return isGameOver
+ int getGameTimer()	get game timer
+ void setGameTimer(int gameTimer)	set game timer
+ House getHouse()	get house
+ BackgroundImage getBackgroundImage()	get background image
+ SunnyBackground getSunnyBackground()	get sunny background image
+ SnowyBackground getSnowyBackground()	get snowy background image
+ RainyBackground getRainyBackground()	get rainy background image
+ int getRedFlowerCount()	get red flower count
+ void setRedFlowerCount(int redFlowerCount)	set red flower count
+ int getRainbowDrakeCount()	get rainbow drake count
+ void setRainbowDrakeCount(int	set rainbow drake count

rainbowDrakeCount)	
+ int getDaffodilCount()	get daffodil count
+ void setDaffodilCount(int daffodilCount)	set daffodil count

3. Package gui

3.1. Package gameBackground

3.1.1. public class SunnyBackground extends Canvas

constructor

+ SunnyBackground(double width, double height)	- call super constructor - clear gc rectangle - all background entities change weather to sunny - set visible to true

3.1.2. public class SnowyBackground extends Canvas

constructor

+ SnowyBackground (double width, double height)	- call super constructor - clear gc rectangle - all background entities change weather to snowy - set visible to true

3.1.3. public class RainyBackground extends Canvas

constructor

+ RainyBackground(double width, double height)	- call super constructor - clear gc rectangle - all background entities change weather to snowy - set visible to true

3.2. Package Map

3.2.1. public interface WeatherChangeable

method

void changeWeather(Config.Weather weather)	abstract method to change weather

3.2.2. public class BackgroundImage extends Entity implements WeatherChangeable

field

- String imagePath	image path for background image

method

+ BackgroundImage()	call super constructor change weather to sunny
+ void changeWeather(Config.Weather weather)	change image path base on weather
+ int getZ()	return -999
+ void draw(GraphicsContext gc)	draw background image in gc by the size of Config.GAMESCREENWIDTH * Config.GAMESCREENHEIGHT

3.2.3. public class Tree extends CollidableEntity implements WeatherChangeable

field

- String imagePath	image path for tree image
- int option	tree image option

constructor

+ Tree(double x, double y, double width, double height, int z, int option)	call super constructor set z value and option value change weather to sunny

method

+ void changeWeather(Config.Weather weather)	change image path base on weather
+ void void draw(GraphicsContext gc)	draw image in gc

3.2.4. public class Bush extends CollidableEntity implements WeatherChangeable

field

- String imagePath	image path for bush image
- int option	bush image option

constructor

+ Bush(double x, double y, double width, double height, int z, int option)	call super constructor set z value and option value change weather to sunny

method

+ void changeWeather(Config.Weather weather)	change image path base on weather
+ void void draw(GraphicsContext gc)	draw image in gc

3.2.5. public class House extends CollidableEntity implements WeatherChangeable

field

- String imagePath	image path for house image

constructor

+ House(double x, double y, double width, double height, int z)	call super constructor set z value change weather to sunny

method

+ void changeWeather(Config.Weather weather)	change image path base on weather
+ void void draw(GraphicsContext gc)	draw image in gc

3.3. Package scene

3.3.1 public class Menu extends GridPane

Fields

- Button sunnyButton	Button to change weather to sunny
- Button snowyButton	Button to change weather to snowy
- Button rainyButton	Button to change weather to rainy
- Button creditButton	Button to display CreditMenu
- Button howToPlayButton	Button to display howToPlayMenu
- Button startButton	Button to start the game
- Button exitButton	Button to exit the game
- GridPane creditMenu	GridPane that displays Credit
- GridPane howToPlayMenu	GridPane that displays how to play the game
- HBox weatherButtonContainer	HBox that contains sunnyButton, snowyButton, rainyButton
- Text titleText	Text that display game's title
- VBox titleAndMenu	VBox that contains titleText and menuButton
- StackPane creditHowtoPlay	StackPane that contains creditMenu and howToPlayMenu

Constructor

+ Menu(PreStory preStory, Stage stage)	- calls setGridSize() - calls initializeTitleText() - calls initializeCreditButton() - calls initializeHowToButton() - calls initializeStartButton(preStory) - calls initializeExitButton(stage) - calls initializeWeatherButtonContainer() - calls initializeTitleAndMenu() - calls setSunnyBackground() - calls initializeCreditHowToPlay(); - Add all components to the grid layout. - Play the RenderableHolder's mainMenuSong with loop.

Method

- void setSunnyBackground()	- Set background to sunny image

	- Set background of sunny icon to #7ca6cc, others icons set background to transparent.
- void setSnowyBackground()	- Set background to snowy image - Set background of snowy icon to #7ca6cc, others icons set background to transparent.
- void setRainyBackground()	- Set background to rainy image - Set background of rainy icon to #7ca6cc, others icons set background to transparent.
- void setGridSize()	set the size of each grid of this
- void initializeTitleAndMenu()	Create VBox that contains titleText and menuButton
- void initializeCreditHowToPlay()	Create StackPane of creditMenu and howToPlayMenu
- void initializeTitleText()	Create a text object with the title "Witch's Garden"
- void initializeSunnyButton()	Create sunnyButton
- void initializeSnowyButton()	Create snowyButton
- void initializeRainyButton()	Create rainyButton
- void initializeWeatherButtonContainer()	Create WeatherButtonContainer which contains sunnyButton, snowyButton, rainyButton
- void initializeExitButton()	Create an "Exit" button
- void initializeStartButton()	Create a "Start Game" button
- void initializeHowToButton()	Create a "How To Play" button
- void initializeHowToMenu()	Create an HowToPlay menu to displays how to play the game and add closeHowToButton
- void initializeCreditButton()	Create a "Credit" button
- void initializeCreditMenu()	Create credit menu to displays credits and add closeCreditButton
- Text getCreditContent()	return the text which is the content of credit

3.3.2. public class EndStory extends GridPane

Fields

- ArrayList<String> ANALOGS	All of story analogs

- int count	Counting of analog
- Text analog	Text analogs displays in each scene
- ImageView storyImage	Image of each scene
- Button nextButton	Button to go to next scene
- VBox textArea	VBox that contains analog

Constructor

+ EndStory(GameEnding gameEnding)	- Initialize a count variable to 0 - Create an ArrayList of String to hold analog texts. - assign ImageView to storyImage - calls initializeAnalogs() , initializeNextButton(gameEnding) , initializeTextArea() - add storyImage , textArea , nextButton to this - set styles of this - Display the first analog message. - if the player reads all dialog Stop any currently playing background music and play the story background music

method

- void showLog(int logCount)	- set analog text to analogs index by count - increase count 1 values
- void setStyle()	set style of this
- void initializeAnalogs()	add string to ANALOGS
- void initializeNextButton(GameEnding gameEnding)	Create a "Next..." button . if count = ANALOGS.size() , calls gameEnding.completeGame().
- void initializeTextArea()	Create a Text object for analog and VBox for displaying text messages.

3.3.3. public class PreStory extends GridPane

Fields

- ArrayList<String> ANALOGS	All of story analogs
- int count	Counting of analog
- Text analog	Text analogs each scene
- ImageView backgroundImage	Background image each scene

- ImageView storyImage	Image of each scene
- Button nextButton	Button to go to next scene
- VBox textArea	VBox that contains analog

Constructor

public EndStory(GameEnding gameEnding)	- Initialize a count variable to 0 - Create an ArrayList of String to hold analog texts. - assign ImageView to storyImage - calls initializeAnalogs() , initializeNextButton(gameStarter) , initializeTextArea() - add storyImage , textArea , nextButton to this - set styles of this - Display the first analog message. - if the player reads all dialog Stop any currently playing background music and play the story background music.

method

+ void showLog(int logCount)	- set analog text to analogs index by count - increase count 1 values
- void initializeAnalogs()	add string to ANALOGS
- void initializeNextButton(GameStarter gameStarter)	Create "Next..." button . if count = ANALOGS.size() , calls gameStarter.startGame().
- void initializeTextArea()	Create a Text object for analog and VBox for displaying text messages.

3.3.4. public class GameEnd extends VBox

Fields

- Button startButton	Button to start the game
- Button exitButton	Button to exit the game
- Text endingText	Text to show at the end of game

Constructor

+ GameEnd(GameStarter gameStarter, Stage stage)	- calls initializeEndingText() , initializeStartButton(GameStarter gameStarter) , initializeExitButton(Stage stage) - set exitButton to exit the game when the exit button is

	clicked.
	- Retrieves the game timer value from the GameController
	  - If timer == 0 : player lose
	  - If timer > 0 : player win
	- Sets the text content of the endingText object

method

- void initializeEndingText()	set Text to endingText
- void initializeStartButton(GameStarter gameStarter)	Create a “Start Game” button. It will call gameStarter.startGame() if this is clicked.
- void initializeExitButton(Stage stage)	Create an “Exit” button. It will call stage.close() if this is clicked.

3.3.5. public class GamePanel extends HBox

Fields

- GameController gameController	gameController which control the game
- GameScreen gameScreen	game screen of the game
- StackPane gameScreenWithEffect	stackpane which contain game screen and effect canvas
- ProgessBar timerBar	show the time remaining in form of bar
- Text clockTimer	show the remaining cooldown time of changing weather
- Text gameModeLabel	Text indicating the current game mode.
- Text redFlowerCount	Text displaying the count of red flowers collected.
- Text rainbowDrakeCount	Text displaying the count of rainbow drakes collected.
- Text daffodilCount	Text displaying the count of daffodils collected.
- Text timerText	Text indicating the timer label.
- int level	Current level of the game.
- Button sunnyButton	Button to change weather to sunny.
- Button snowyButton	Button to change weather to snowy.

- Button rainyButton	Button to change weather to rainy.
- VBox targetVeggieContainer	Container for displaying target vegetables.
- HBox targetVeggie	Container for displaying individual target vegetables.

Constructor

- GamePanel(GameController gameController, GameScreen gameScreen, StackPane gameScreenWithEffect, int level)	- Initializes timer text, game mode label, clock timer, weather buttons, target veggies, and their container. - Sets the style and adds elements to the layout.

method

- void addElement()	- Adds elements to the GamePanel layout. - Creates and arranges timer label and bar, game mode label, and weather buttons.
- void setStyle()	Sets the style for the GamePanel. - Sets alignment, size, spacing, and background color.
- void initializeClockTimer()	Initializes the clock timer text. - Sets the font, size, and color.
- void initializeTargetVeggie()	Initializes the target vegetable icons and counts. - Loads vegetable icons and sets their properties. - Initializes and arranges count texts.
- void initializeVeggieCount()	Initializes the count texts for vegetables. - Sets font, size, and color.
- void initializeTargetVeggieContainer()	Initializes the container for displaying target vegetables. - Sets title font, size, and color. - Arranges target vegetables within the container.
- void initializeTimerText()	Initializes the timer label and bar. - Sets font, size, color, and style for the timer text. - Sets the width and style for the timer bar.
- void initializeGameModeLabel()	Initializes the game mode label. - Determines the potion name based on the level.

	- Sets font, size, and color for the label.
- void initializeWeatherButton()	Initializes the weather change buttons. - Sets style and action for each button.
- void updateTimerBar(double time)	Updates the timer bar progress based on the given time.
- void updateClockTimer()	Updates the clock timer text based on the game controller's clock.
- void updateVeggieCount()	Updates the counts of collected vegetables.
- void handleSunnyButton()	Handles the action when the sunny weather button is pressed. - Changes the season to sunny if possible. - Plays clock sound and updates background accordingly.
- void handleSnowyButton()	Handles the action when the snowy weather button is pressed. - Changes the season to snowy if possible. - Plays clock sound and updates background accordingly.
- void handleRainyButton()	Handles the action when the rainy weather button is pressed. - Changes the season to rainy if possible. - Plays clock sound and updates background accordingly. - Creates and adds special effect canvas for rainy weather.

3.3.6. public class GameScreen extends Canvas

Constructor

+ GameScreen(double width, double height)	- calls super(width, height) - set visible to true - calls addListener()

method

- void addListener()	adds event listeners for key presses and releases to the canvas.
- void paintComponent()	- clears the canvas - Iterates over all entities stored in the

	RenderableHolder class. If the entity is visible and not destroyed, it calls the draw() method of the entity

3.4. Package WeatherCanvas

3.4.1 public class RainyCanvas extends Canvas

field

- List<Raindrop> raindrops	List to store raindrop objects.

Constructor

+ RainyCanvas()	Constructs a new RainyCanvas. - Initializes the canvas with specified dimensions. - Generates initial raindrops. - Starts an animation timer to update and render raindrops.

method

- void updateRaindrops()	Updates the positions of all raindrops. - Iterates through the raindrops list and updates their positions.
- void render()	Renders all raindrops on the canvas. - Clears the canvas. - Renders each raindrop using its render method.

Inner class

3.4.1.1 private static class RainDrop

field

- double x	X-coordinate of the raindrop.
- double y	Y-coordinate of the raindrop.
- double speed	Speed of the raindrop.

Constructor

+ Raindrop()	Constructs a new Raindrop. - Initializes the raindrop with random position and speed.

method

- void update()	Updates the position of the raindrop. - Moves the raindrop downwards. - Resets the raindrop position if it reaches the bottom of the canvas.
- void render(GraphicsContext gc)	Renders the raindrop on the canvas. - Fills a rectangle representing the raindrop on the specified graphics context.

3.4.1 public class SnowyCanvas extends Canvas

field

- List<Snowflake> snowflakes	List to store snowflake objects.

Constructor

+ SnowyCanvas()	Constructs a new SnowyCanvas. - Initializes the canvas with specified dimensions. - Generates initial snowflakes. - Starts an animation timer to update and render snowflakes.

method

- void updateSnowflakes()	Updates the positions of all snowflakes. - Iterates through the snowflakes list and updates their positions.
- void render()	Renders all snowflakes on the canvas. - Clears the canvas. - Renders each snowflake using its render method.

Inner class

3.4.1.1 private static class Snowflake

field

- double x	X-coordinate of the snowflake.
- double y	Y-coordinate of the snowflake..
- double speed	Speed of the snowflake.
- double size	Size of the snowflake.

Constructor

+ Snowflake()	Constructs a new Snowflake. - Initializes the snowflake with random position, speed, and size.

method

- void update()	Updates the position of the snowflake. - Moves the snowflake downwards. - Resets the snowflake position if it reaches the bottom of the canvas.
- void render(GraphicsContext gc)	void render(GraphicsContext gc) | Renders the snowflake on the canvas. - Fills an oval representing the snowflake on the specified graphics context.

3.4.3. public class WhiteCanvas extends Canvas

field

- final GraphicsContext gc	graphic context
- double alpha	init alpha value = 0
- final double minSpeed	init min spead = 0.1
- final double maxSpeed	init max spead = 0.5
- double speed	white canvas speed
- int level	level of canvas
- double ratioTime	ratio time of canvas
- final double speedScaleFactor	speed scale factor of canvas

constructor

+ WhiteCanvas(double width, double height, int level, double ratioTime)	- call super constructor - set level, ratio time, min speed and gc - call adjustSpeed()

method

+ void start()	Create an AnimationTimer that - If lastUpdate = 0, init lastUpdate to the now and return. - Calculate the elapsed time - Update the alpha value - Clear the canvas. - Fill the canvas with a white color with adjusted transparency - Update lastUpdate to the current time (now). Run th
+ void adjustSpeed()	- Calculate the max additional speed - Scale the speedFactor using speedScaleFactor - Update the speed to speed + speedFactor

4. Package guiSharedObject

4.1. public class Entity

field

# double x	entity's x position
# double y	entity's y position
# int z	entity's z position
# boolean visible	entity's boolean
# boolean destroyed	entity's destroyed

constructor

# Entity(double x,double y)	set visible true, set destroyed false set position x and position y
# Entity()	set visible true, set destroyed false

method

+ double getX()	get entity’s x position
+ void setX(double x)	set entity’s x position
+ double getY()	get entity’s y position
+ void setY(double y)	set entity’s y position
+ boolean isDestroyed()	get is entity is destroyed
+ boolean isVisible()	get entity visible status
+ int getZ()	set entity’s z position
+ void draw(GraphicsContext gc)	draw entity on map

4.2. public class CollidableEntity extends Entity

field

# boolean width	entity’s width
# boolean height	entity’s height

constructor

# CollidableEntity()	call super constructor

method

+ boolean collideWith(double otherX,double otherY,double otherWidth,double otherHeight)	check if an entity collides with another object by checking top, right, right, bottom of two entities.
+ double getWidth()	get entity’s Width
+ void setWidth(double width)	set entity’s Width
+ double getHeight()	get entity’s Height position

4.3. public class InputUtility

field

# final ArrayList<KeyCode> keyPressed	array for keys that user pressed

method

+ boolean getKeyPressed(KeyCode keycode)	return if keypressed contain keycode
+ void setKeyPressed(KeyCode keycode,boolean pressed)	set keypressed by add or remove keycode
+ void clearKeyPressed()	clear key in keypressed

4.4. public class RenderableHolder

field

- RenderableHolder instance = new RenderableHolder()	renderable holder instance
- List<Entity> backgroundEntities	list of background entities
- List<Entity> entities	list of entities
- Comparator<Entity> comparator	comparator for entity
- Image witchSprite	witch Sprite
- Image witchWalkSprite	witch Walk Sprite
- Image witchBroomSprite	witch Broom Sprite
- Image witchWalkBroomSprite	witch Walk Broom Sprite
- Image witchAttackSprite	witch Attack Sprite
- Image witchRightSprite	witch Right Sprite
- Image witchRightBroomSprite	witch Right Broom Sprite
- Image witchLeftSprite	witch Left Sprite
- Image witchLeftBroomSprite	witch Left Broom Sprite
- Image broomSprite	broom Sprite
- Image normalSlimeSprite	normal Slime Sprite
- Image hitHardSlimeSprite	hit Hard Slime Sprite
- Image speedSlimeSprite	speed Slime Sprite
- Image rainbowDrakeIdleSprite	rainbow Drake Idle Sprite
- Image redFlowerIdleSprite	red Flower Idle Sprite

- Image daffodilIdleSprite	daffodil Idle Sprite
- MediaPlayer hitSound	hit Sound
- MediaPlayer collectSound	collect Sound
- MediaPlayer clockSound	clock Sound
- MediaPlayer mainMenuSong	main Menu Song
- MediaPlayer storySong	story Song
- MediaPlayer gameSong	game Song

constructor

+ RenderableHolder()	init entities, background entity, and comparator

method

+ void loadResource()	load image and MediaPlayer resource
+ void add(Entity entity)	add entity to entities list
+ void addBackground(Entity entity)	add entity to background entities list
+ void update()	remove destroyed entity
+ RenderableHolder getInstance()	get RenderableHolder instance
+ List<Entity> getEntities()	get entities list
+ List<Entity> getBackgroundEntities()	set entities list

5. Main

5.1. public class Main extends Application

field

- int level = 1	level of game

method

+ void main(String[] args)	Launch the JavaFX application
+ void start(Stage primaryStage)	- Create a Menu scene by new Manu() - Set the scene on the primaryStage.

	- Load and set the icon for the stage. - Set the title of the window to "Witch's Garden". - Make the window non-resizable. - Show the window. - Set the action to terminate the application when the window is closed.
- void preStory(Stage primaryStage)	- Create a preStory Scene by new PreStory() - Set the scene on the primaryStage.
- void endStory(Stage primaryStage)	- Create a preStory Scene by new EndStory() - Set the scene on the primaryStage.

- void startGame(Stage primaryStage)	- create a waiting scene
	
	- get the gameController instance
	- clear gameControlle stats and init game
	- Create and configure the game screen and background effects.
	- Create a game panel with the game screen and background effects.
	- Add the game panel and game screen to the root and Set the scene on the primaryStage
	
	- Stop any currently playing background music and start the game song with a loop.
	
	- create AnimationTimer that will loop to
	  - Check if the game is over, if so, stop the timer and transition to the game end scene.
	  - update and render game components
	  - Paint the game screen.
	  - Update the clock timer, timer bar, and veggie count in the game panel.
	  - Call GameController.play()
	  - Update the RenderableHolder instance.

- void gameEnd(Stage primaryStage)	- check the gameController timer
	  - If the game timer is not zero, increase the level by 1.
	  - If the level reaches 4, call EndStory(primaryStage).
	  - Otherwise, restart the game by calling startGame(primaryStage).
	- Create a new Scene with the GameEnd instance and set it on the primaryStage.

- void completeGame(Stage primaryStage)	- create completeGame scene with congratulation text and exit button