Plants VS Zombies
Documentation

Created by

Akkharawat Burachokviwat 6330585221

Thanaphum Thepwan 6330223021

2110215 Programming Methodology

Semester 1 Year 2021

Chulalongkorn University

Plants Vs Zombies

1. Introduction

Get ready to soil your plants because a mob of zombies is going to invade your house. Use your plants as weapons to fight, which consist of Sunflower, Peashooters, Wallnuts, Cherrybombs, ChilliPepper and Repeater to overcome 3 types of zombies before they break down your house.

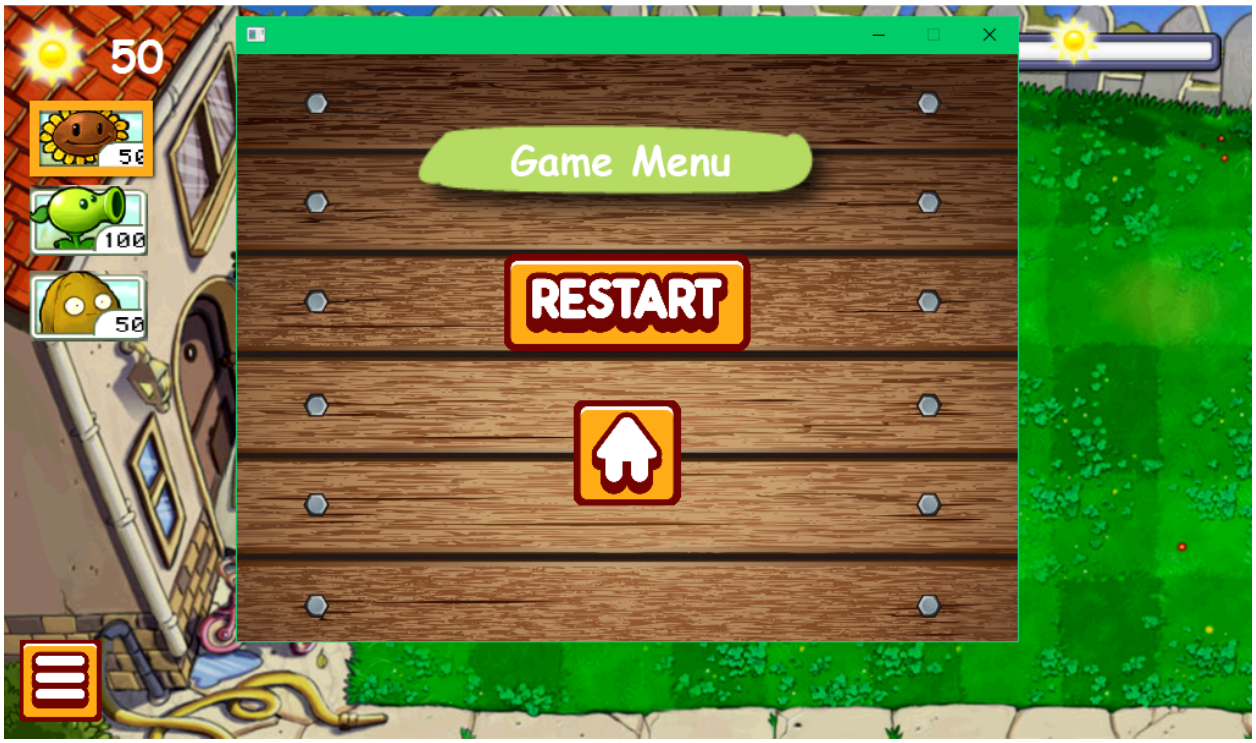
2. Rules

At the beginning of the game, player has 50 sun counts which can plant only Sunflower. Player should try to collect sun from Sunflower or the sun that is floating down (only in day mode), each sun is equal to 25 sun counts. Sun count is used to exchange plants for planting in order to fight zombies. When you soil the plant, that plant will lock for a while then it will become available again. Plants have theirs own ability such as shoot zombie, retard zombie, produce sun. Player can uproot the plant by using a shovel. Player will win when the progress bar is full. However, player will lose when one of the zombies reaches the house.

3. Example

Figure below shows 3 types of plants on the lawn which consist of PeaShooter, Sunflower and Wallnut. These plants are fighting with zombies that are heading to the house. Sun count label is located on the top-left which is 0. Plants that players can choose to plant located on the left side. When picking a plant, player can drop anywhere on the lawn grid. Top-right is the progress bar which is shown as 0. To the left of the progress bar is a shovel which is used to uproot the plant.

Player can click the three-dashes button to select the menu.



In case that player wins the level, then the game shows the plant that player can use in the next level, so in the last level (level 5) player can choose to play the next level or go to the main menu by clicking the button located on bottom-right.

In case that player loses the level, player is urged to play such level again.

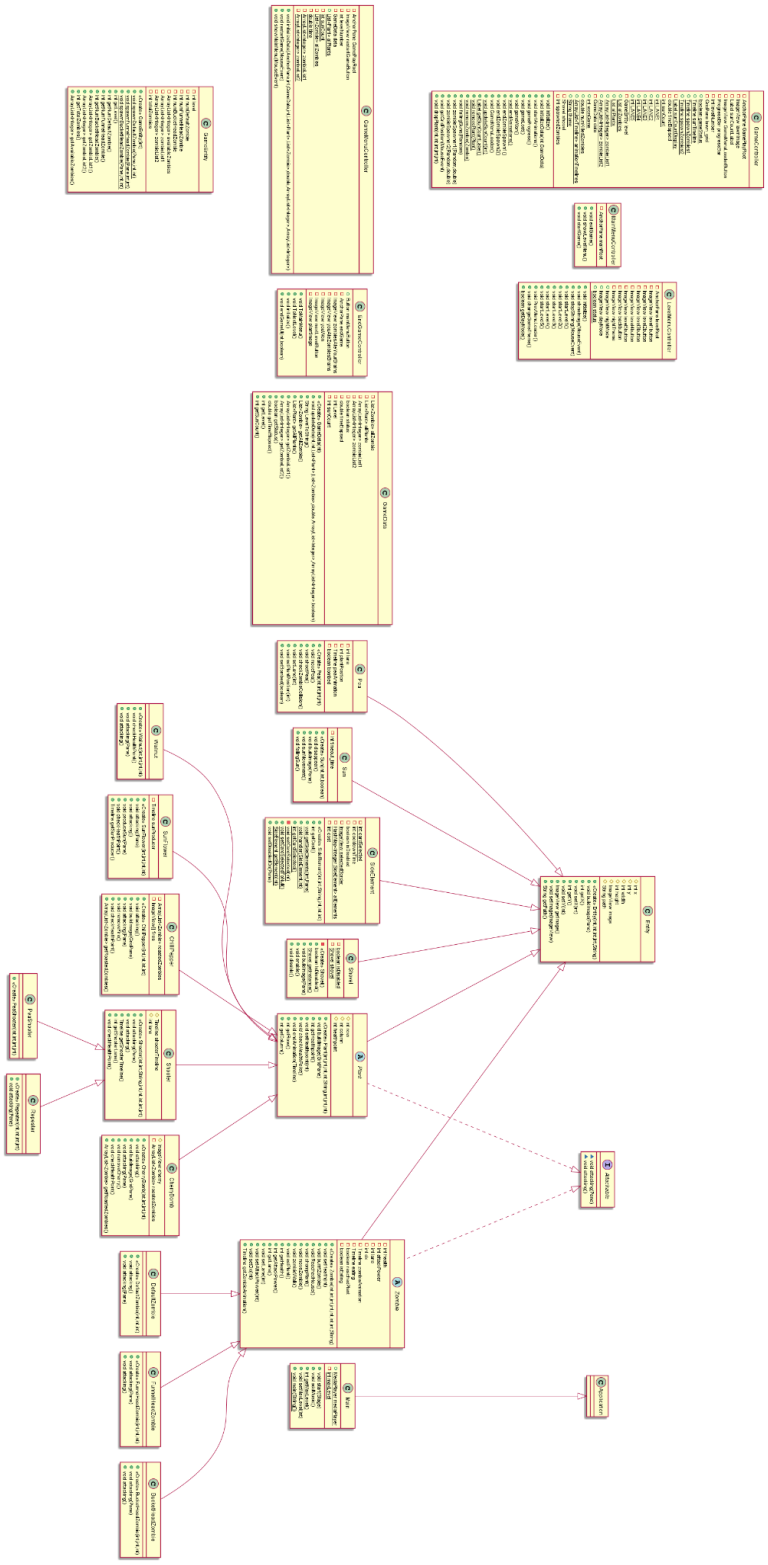
4. Scene

Click PLAY to proceed to the game stage. Otherwise, click EXIT to close the game.

Choose a level to play or go back to the main page.

Playing in the night theme.

5. Class diagram



6. Class Details

6.1. Package Entity

6.1.1 Class Sun

This class represents the Sun entity which the Player can click to collect sun points.

6.1.1.1 Field

Name	Description
- final int timeout_time	- The time for the sun to be on the screen.

6.1.1.2 Constructor

Name	Description
+ Sun(int x, int y, boolean isFalling)	- Initialize super class with given x, y and set width = 50, height = 50 and set path to the sun picture. - Initialize timeout_time. - Using disappear() method to make the sun disappear when timeout.

6.1.1.3 Methods

Name	Description
+ void disappear()	Using thread: - Make the sun disappear when timeout.
+ void buildImage(Pane pane)	- Set image on pane. - Set an event when the player clicked on the image. (the sun will be disappeared and increase sun point by 25)
+ void sunMovement()	- The direction for the sun to move.
+ void fallingSun()	- Falling animation of the sun.

6.1.2 *Abstract Class Zombie*

This class is the base class for Zombie, used as the villain of the game. This class ensures that every type of Zombie has enough methods to work.

6.1.2.1 Fields

Name	Description
- int health	- Health points of Zombie
- int attackPower	- Attack power of Zombie
- int lane	- Position of Zombie as a lane.
- int dx	- Amount of pixels that Zombie walks.
- Timeline zombieAnimation	- Animation of zombies such as walking, dying.
- Timeline eating	- Zombie eating animation.
- boolean reachedPlant	- Check that Zombies reach the plant.
- boolean isEating	- Check that Zombies are eating plants.

6.1.2.2 Constructor

Name	Description
+ Zombie(int health, int attackPower, int x, int y, int width, int height, int lane, String path)	- Initialize all fields. - Set dx to -1.

6.1.2.3 Methods

Name	Description
+ setHealthPoint(int health)	- Set health by the given parameter. - If health less than or equal to 0: ● Increase numZombiesKilled in GamePlayController by 1. ● setVisible of image to false. ● setDisable of image to true. ● stop zombieAnimation. ● If a zombie is eating, stop it. Otherwise, do nothing. ● Make yuck sound for this zombie by using MediaPlayer. ● Remove this zombie from allZombies in GamePlayController. - If health less than or equal to 7: ● change image of zombie to

	normalzombie. ● set image FitHeight to 65. ● set image FitWidth to 115. ● set width to 65. ● set height to 115.
+ void burntZombie()	- Set image of zombie to burntZombie. - Set image FitHeight to 115. - Set image FitWidth to 65. - Set dx to 0. - Set health to 0. - Stop the eating animation. - Increase numZombiesKilled in GamePlayController by 1. - use thread sleep 5000 ms and set image Visible to false and set image Disable to true.
+ void ReachedHouse()	- If x position of image less than or equal to 220: ● Play brainzSound using MediaPlayer. ● set AutoPlay to true. ● set wonGame to -1.
+ void chompPlant()	- Play chompSound using MediaPlayer. - set AutoPlay to true. - set StartTime by Duration.seconds(0). - set StopTime by Duration.seconds(1). - set CycleCount to 1.
+ void moveZombie()	- Initialize Timeline animation using KeyFrame and set Duration to 70 millis, and use method zombieWalk. - set CycleCount of animation to Indefinite - set zombieAnimation to this Timeline animation. - add this animation to animationTimelines in GamePlayController.
+ void zombieWalk()	if x position of zombie more than 220 and health more than 0: ● set x to x + dx. ● Using try-catch with method eatPlant() and catch java.util.ConcurrentModificationException, then call method ReachedHouse().
+ void eatPlant()	- Using Collections.synchronizedList in GamePlayController (in order to be sure that each method call has to fully run and return before the other one could run). - Initialize boolean foundPlant to false. - Using iterator to iterate in allPlants. - if plant row same as zombie lane to following description, otherwise set dx to -1: ● If the difference between zombie and

	plant is less than or equal 50: ○ set foundPlant to true. ○ set reachedPlant to true. ○ set isEating to true. ○ Initialize Timeline eat using KeyFrame and set Duration to 1000 millis, and use method chompPlant(). ○ set Timeline CycleCount to 1000. ○ set eating field to eat. ○ add this animation to animationTimelines in GamePlayController. ○ set isEating to false. ○ set plant health to plant health minus zombie attackPower. ○ if plant.getHealth() less than or equal to 0: ■ set plant health to 0. ■ remove plant from allPlants in GamePlayController. ■ set plant image Visible to false. ■ set plant image Disable to true. ■ set dx to -1. ■ set reachedPlant to false. ■ Stop eating animation. ● Otherwise: ○ set dx to -1. ○ set reachedPlant to false. ○ Stop eating animation - If foundplant is false: ● set dx to -1. ● set reachedPlant to false. ● Stop eating animation.
getter and setter	getter and setter.

6.1.3 *Abstract Class Plant*

This class is the base class for Plant, used as the Hero of the game. This class ensures that every type of Plant has enough methods to work.

6.1.3.1 Fields

Name	Description
# int row	- Row of plants.
# int column	- Column of plants.

# int healthpoint	- Plant's healthpoint.

6.1.3.2 Constructor

Name	Description
+ Plant(int x, int y, int width, int height, String path, int healthpoint, int column, int row)	- Initialize super class with given x, y, width, height, path. - Set healthpoint for plant. - Set column for plant. - Set row for plant.

6.1.3.3 Methods

Name	Description
+ void buildImage(GridPane lawn)	- Set image for plant and add to lawn.
+ void endAnimation(Timeline timeline)	- Stop the animation of timeline.
+ void setHealthpoint(int healthpoint)	- Set health of the plant. - if health is less than or equal to zero, then set the image visible to false and disable to true.
+ void checkHealthPoint()	- This method does nothing.
getter and setter	- getter and setter for class fields.

6.1.4 Class Pea

This class is Entity represents Pea that came out from shooter like a bullet and will attack the zombies.

6.1.4.1 Fields

Name	Description
# int lane	- Lane for Pea.
# int plantPosition	- Position for plant.
- Timeline peaAnimation	- Animation for pea.
- boolean bombed	- It will become true, if the pea is going to hit a zombie.

6.1.4.2 Constructor

Name	Description
+ Pea(int x, int y, int plantPosition, int lane)	- Initialize super class with given x, y and set width = 20, height = 20 and set image path for pea.

	- Set plantPosition. - Set lane for Pea. - Set bombed to false.

6.1.4.3 Methods

Name	Description
+ void movePea()	- set x to x+1. - check for collisions using the method checkZombieCollision.
+ void shootPea()	- In lane animation for Pea. (set KeyFrame duration to Duration.millis(5)) - add the animation to GameController.animationsTimelines.
+ void checkZombieCollision()	- If Pea hits the zombie, Pea will disappear and make a splat sound.
setters	setter of fields.

6.1.5 Class Shovel

This class is Entity represents the shovel. The shovel is used to destroy plants. Moreover, this class is a singleton class.

6.1.5.1 Fields

Name	Description
- boolean isDisabled	- It's true by default.
- Shovel shovel	- Shovel.

6.1.5.2 Constructor

Name	Description
- Shovel()	- Initialize super class with x = 680, y = 10, width = 58, height = 58 and set image path for shovel.

6.1.5.3 Methods

Name	Description
+ boolean isDisabled	- Return status of shovel.
+ Shovel getInstance()	- Return shovel. If no shovel initialized, initialize and return a new one.
+ buildImage(Pane p)	- Use buildImage from super class. - Set on click action.

+ void enable()	- Set the image to have glow effects when enabled.
+ void disable()	- Remove glow effects.

6.1.6 Class SideElement

This class is Entity, and deals with the all side card element. Player must click a card to plant.

6.1.6.1 Fields

Name	Description
- int cardSelected	- It's -1 by default and will change depending on which card is selected.
- int cooldownTime	- Card cool down time.
- boolean isDisabled	- It's false by default but it will be true if the card is cooldown.
- ImageView selectedBorder	- The Border image. If Player clicks on a card the border will display.
- HashMap<Integer, SideElement> allElements	- Using map data structure to store data. The key type is Integer, mapped value is SideElement.
- final int cost	- Cost for selected card.

6.1.6.2 Constructor

Name	Description
+ SideElement(int x, int y, String path, int width, int height, int cost)	- Initialize super class with given x, y, width, height, path. - Initialize variable cost with given cost.

6.1.6.3 Methods

Name	Description
+ getCost()	- Getter for cost.
+ void getSideElements(int level, Pane pane)	- Initialize allElements field. - This method will build image on the pane depending on which level that Player is playing. ● If on level 1 or more. It has sunflowers and peashooters. ○ Initialize the SideElement sunflowerCard with x = 24, y = 79, sunflowerCard imagepath, cost =50, width = 97, height =

	58. ○ set sunflowerCard cooldown to 5000. ○ add sunflowerCard to allElements with key = 1, value = this sunflowerCard. ○ set handler when clicked on sunflowerCard using method handler. ○ Initialize the SideElement peaShooterCard with x = 22, y = 147, peashooterCard image path, cost = 100, width = 97, height = 58 ○ set peaShooterCard cooldown to 6000. ○ add peaShooterCard to allElements with key = 2, value = this peaShooterCard. ○ set handler when clicked on peaShooterCard using method handler. ● If on level 2 or more, wallnut is added to the game. ○ Initialize the SideElement wallnutCard with x = 22, y = 217, wallnutCard imagepath, cost =50, width = 97, height = 58. ○ set wallnutCard cooldown to 7000. ○ add wallnutCard to allElements with key = 3, value = this wallnutCard. ○ set handler when clicked on wallnutCard using method handler. ● If on level 3 or more, cherrybomb is added to the game. ○ Initialize the SideElement cherrybombCard with x = 22, y = 284, cherrybombCard imagepath, cost = 150, width = 97, height = 58. ○ set cherrybombCard cooldown to 15000. ○ add cherrybombCard to allElements with key = 4, value = this cherrybombCard. ○ set handler when clicked on cherrybombCard using method handler. ● If on level 4 or more, repeater is added to the game. ○ Initialize the SideElement repeaterCard with x = 23, y =

	352, repeaterCard imagepath, cost = 200, width = 97, height = 58. ○ set repeaterCard cooldown to 10000. ○ add repeaterCard to allElements with key = 5, value = this repeaterCard. ○ set handler when clicked on repeaterCard using method handler. ● If on level 5 or more, chillipepper is added to the game. ○ Initialize the SideElement chillipepperCard with x = 24, y = 420, chillipepperCard imagepath, cost = 125, width = 97, height = 58. ○ set repeaterCard cooldown to 12000. ○ add chillipepperCard to allElements with key = 6, value = this chillipepperCard. ○ set handler when clicked on chillipepperCard using method handler. - Initialize selectedborder with selectedborder path and set Visible to false, set Disable to true.
+ void hander(SideElement sideElement, int cardSelected)	- If sideElement is not disabled, set select on this card using method SetCardSelected. - Disable the Shovel.
+ int getCardSelected()	- return the key value of the card.
- void setCardSelected(int i)	- set cardselected to i. - set selectedborder Visible to true. - set selectedborder on this card.
+ void setCardSelectedToNull()	- set cardselected to -1. - set selectedborder Visible to false.
+ SideElement getElement(int x)	- return SideElement from allElement.
+ void setDisabledOn(Pane pane)	- set isDisabled to true. - Initialize lock image to display on the card. - set the lock to display until out of cooldown time.

6.2. Package Entity.zombie

6.2.1 Class DefaultZombie

This class represents normal zombies which have less health and attack power.

6.2.1.1 Constructor

Name	Description
+ DefaultZombie(int x, int y, int lane)	- Initialize super class with given x, y, lane, health = 5, attackPower = 2, width = 68, height = 118, and path of the default zombie image.

6.2.1.2 Methods

Name	Description
+ void attacking(Pane pane)	-This override method does nothing.
+ void attacking()	- Use the method eatPlant() from super class.

6.2.2 Class BucketHeadZombie

This class represents zombies with high health and high attack power.

6.2.2.1 Constructor

Name	Description
+ BucketHeadZombie(int x, int y, int lane)	- Initialize super class with given x, y, lane, health = 22, attackPower = 4, width = 65, height = 120, and path of the bucket head zombie image.

6.2.2.2 Methods

Name	Description
+ void attacking(Pane pane)	- This override method does nothing.
+ void attacking()	- Use the method eatPlant() from super class.

6.2.3 Class FunnelHeadZombie

This class represents zombies with medium health and medium attack power.

6.2.3.1 Constructor

Name	Description
+ FunnelHeadZombie(int x, int y, int lane)	- Initialize super class with given x, y, lane, health = 12, attackPower = 2, width = 134, height = 124, and path of the funnel head zombie image.

6.2.3.2 Methods

Name	Description
+ void attacking(Pane pane)	- This override method does nothing.
+ void attacking()	- Using the method eatPlant() from super class.

6.3. Package Entity.plant

6.3.1 Class CherryBomb

This class represents cherry bomb which is a plant that can attack zombie.

6.3.1.1 Field

Name	Description
# ImageView cherry	- image of cherry
- ArrayList<Zombie> roastedZombies	- List of roasted zombie

6.3.1.2 Constructor

Name	Description
+ CherryBomb(int x, int y, int column, int row)	- Initialize super class with given x, y, row, column, healthpoint = 4, width = 90, height = 68, and path of the cherry bomb image.

6.3.1.3 Methods

Name	Description
+ void attacking()	- this override method does nothing.

+ void buildImage(GridPane lawn)	- buildImage by super class. - set cherry to powie image. - set FitHeight to 180. - set FitWidth to 160. - set x to x - 40. - set y to y - 20. - set Visible to false. - set Disable to true. - initialize roastedZombies field.
+ void attacking(Pane pane)	- add a cherry image to the pane. Using thread with thread sleep 1800 ms: - create mediaPlayer with cherry bomb sound. - set image Visible to false. - set image Disable to true. - set cherry image Visible to true. - iterate all zombies and if zombie x position is less than or equal to x + 250 and more than or equal to x - 150: ● add this zombie to roastedZombie. ● call method burntZombie(). - remove this plant in GamePlayController.allPlants. - remove all roasted zombies in GamePlayController.allZombies. - call method removeCherry().
+ void removeCherry()	Using thread with thread sleep 1250 ms: - set cherry Visible to false. - setHealthpoint to 0.
+ void checkHealthPoint()	- this override method does nothing.
+ getter of ArrayList<Zombie> getRoastedZombies()	- return roastedZombies.

6.3.2 Class ChilliPepper

This class represents chili pepper which is a plant that can blast the zombies.

6.3.2.1 Field

Name	Description
- ImageView[] fires	- List of images of chilli pepperfire.
- ArrayList<Zombie> roastedZombies	- List of roasted zombies.

6.3.2.2 Constructor

Name	Description
+ ChilliPepper(int x, int y, int column, int row)	- Initialize super class with given x, y, row, col, healthpoint = 4, width = 100, height = 100,

	and path of the chillipepper image. - initialize fires with size 9.

6.3.2.3 Methods

Name	Description
+ void attacking()	- this override method does nothing.
+ void buildImage(GridPane lawn)	- Use buildImage from super class. - initialize imageView and set Height to 100 and Width to 100. - set imageView Disable to true. - set imageView Visible to false. - add this image to lawn pane. - create 9 images that have properties above and add them to fires.
+ void attacking(Pane pane)	Using thread with thread sleep 1650 ms: - set image Visible to false. - set image Disable to true. - create mediaPlayer with jalapeno sound. - set Visible of all images in fires to true. - iterate all zombies and if row of this plant is same as lane of zombie: ● call method burntZombie(). - remove this plant in GamePlayController.allPlants. - remove all roasted zombies in GamePlayController.allZombies. - call method removeFire().
+ void removeFire()	Using thread with thread sleep 1500 ms: - set Visible of all images in fires to false. - set Disable of all images in fires to true. - setHealthpoint to 0.
+ void checkHealthPoint()	- this override method does nothing.
+ getter of ArrayList<Zombie> getRoastedZombies()	return roastedZombies.

6.3.3 Class Shooter

This class is the plant. It is the base class for all shooters.

6.3.3.1 Fields

Name	Description
# Timeline shooterTimeline	- Timeline for Shooter animation.
# int lane	- Shooter's lane.

6.3.3.2 Constructor

Name	Description
+ Shooter(int x, int y, String path, int healthpoint, int width, int height, int column, int row)	- Initialize super class with given parameters. - Set shooter lane to row.

6.3.3.3 Methods

Name	Description
+ void attacking(Pane pane)	- Initialize a new shooter's Timeline with Keyframe duration to 2 seconds, and add a handler when a zombie is coming in the lane, it will start Pea animation (set the start position by x+50 and y+25). - set animation CycleCount to Timeline.INDEFINITE. - Add shooter's Timeline to animationTimelines.
+ void attacking()	- This override method does nothing.
+ void checkHealthpoint()	- End the shooter's animation when the shooter's healthpoint is less than 0.
getters	- getter for all fields

6.3.4 Class PeaShooter

This class represents normal shooters.

6.3.4.1 Constructor

Name	Description
+ PeaShooter(int x, int y, int column ,int row)	- Initialize super class with given x, y, row, column and set healthpoint = 100, width = 60, height = 62 and path of image file.

6.3.5 Class Repeater

This class represents a repeater, the shooter that shoots the Pea twice at a time.

6.3.5.1 Constructor

Name	Description
+ Repeater(int x, int y, int column ,int row)	- Initialize super class with given x, y, row, column and set healthpoint = 150, width = 60, height = 62 and path of image file.

6.3.5.2 Methods

Name	Description
+ void attacking(Pane pane)	- Initialize a new shooter's Timeline with Keyframe duration to 2 seconds, and add a handler when a zombie is coming in the lane, it will start Pea animation twice. - Add shooter's Timeline to animationTimelines.

6.3.6 Class Wallnut

This class represents walnut, the defensive object of the game.

6.3.6.1 Constructor

Name	Description
+ Wallnut(int x, int y, int column ,int row)	- Initialize super class with given x, y, row, column and set healthpoint = 400, width = 60, height = 75 and path of image file.

6.3.6.2 Methods

Name	Description
+ void checkHealthPoint()	- If healthpoint is equal or less than 0, set Visible to false, set Disable to true and remove it from the allPlants.
+ void attacking(Pane pane)	- This override method does nothing, because wallnut can't attack zombie.
+ void attack()	- This override method does nothing, because wallnut can't attack zombie.

6.3.7 Class Sunflower

This class represents sun flower bomb which is a plant that give player sun to increase sun count.

6.3.7.1 Fields

Name	Description
- Timeline sunProducer	- Sunflower animation.

6.3.7.2 Constructor

Name	Description
+ SunFlower(int x, int y, int column, int row)	- Initialize super class with given x, y, row, column, healthpoint = 100, width = 73, height = 74, and path of the sun flower image.

6.3.7.3 Methods

Name	Description
+ void attacking(Pane pane)	- call method produceSun(pane).
+ void produceSun(Pane pane)	- use Timeline to produce sun and make the sun shine. - add Timeline to animationTimelines.
+ void attacking()	- this override method does nothing.
+ void checkHealthPoint()	- If the sunflower is dead,then end the animation.
+ Timeline getSunProducer()	- return sunProducer.

6.4. Package Entity.base

6.4.1 Interface Attackable

This class is the interface for Entities in the game that have the ability to attack.

6.4.1.1 Methods

Name	Description
+ void attacking()	- Normal attacking.
+ void attacking(Pane pane)	- Normal attacking which makes changes on the pane.

6.4.2 Class Entity

This class is the base class for all Entities in the game. This class ensures that every Entity has enough methods to work.

6.4.2.1 Field

Name	Description
# int x	- Horizontal position in the game.

# int y	- Vertical position in the game.
# int width	- Width of sprite image.
# int height	- Height of sprite image.
# ImageView image	- Image of sprite.
# String path	- String url of image.

6.4.2.2 Constructor

Name	Description
+ Entity(int x, int y, int width, int height, String path)	- Initialize all fields

6.4.2.3 Methods

Name	Description
+ void buildImage(Pane pane)	- Set the image with the ImageView by create an Image that path of the image and set ImageView fit width to width and fit height to height, as well as set x and y coordinate by field x and y. Then, set the image on the pane.
getter and setter	getter and setter of fields x, y, image, path.

6.5. Package logic

6.5.1 Class GameController

This class is the game system. Most of the game's global variable are kept here

6.5.1.1 Fields

Name	Description
- AnchorPane GamePlayRoot	- main root when play game.
- ImageView lawnImage	- lawn image.
- Label sunCountLabel	- numbers of sun count
- ProgressBar progressBar	- progress bar
- int levelNumber	- level
- GridPane lawn_grid	- grid pane based on a lawn image..

+ boolean gameStatus	- Night mode's gameStatus is false, day mode's is true.
+ Timeline sunTimeline	- Sun animation (sun rise, sun drop).
+ Timeline spawnZombies1	- Spawn zombie in a specific lane.
+ Timeline spawnZombies2	- Spawn zombie in a specific lane.
- Label sunCountDisplay	- Number of sun points that player stored.
- double timeElapsed	- Time countdown which shows by progress bar when playing the game.
- int sunCount	- Number of sun points that player stored.
+ final int LANE1 = 50	- y position for lane 1.
+ final int LANE2 = 150	- y position for lane 2.
+ final int LANE3 = 250	- y position for lane 3.
+ final int LANE4 = 350	- y position for lane 4.
+ final int LANE5 = 450	- y position for lane 5.
+ List allZombies	- List of all zombies in the game.
+ List allPlants	- List of all plants in the game.
+ ArrayList<Integer> zombieList1	- Store available zombies.
+ ArrayList<Integer> zombieList2	- Store available zombies.
+ int wonGame	- If wonGame is 0 player wins, otherwise player loses.
+ int numKilledZombies = 0	- Number of killed zombies.
+ArrayList<Timeline> animationTimelines	- Stored all animations in this class.
+ String theme = "day"	- The game theme.
- GameEntity level	- The game level.
- GameData data	- most entity data.
- int spawnedZombies = 0	- Number of zombies spawned.
- Shovel shovel	- Shovel used to uproot the plant.

6.5.1.2 Methods

Name	Description
+ void initialize()	- Create mediaPlayer sound. - Set mediaPlayer duration to 5 seconds. - setAutoPlay to true. - Initialize allPlants and allZombies fields. - Set the gameStatus to true. - set sunCountDisplay to sunCountLabel.
+ void initializeData(int level, GameData gameData)	- Set wonGame = 0. - Set zombieList1 to GameData.getZombieList1(). - Set zombieList2 to GameData.getZombieList2(). - Set allPlants to GameData.getAllPlants(). - Set allZombies to GameData.getAllZombie(). - sunCount to GameData.getSunCount() - Set timeElapsed to GameData.getTimeElapsed(). - set levelNumber = level - set level to GameEntity(level) - initialize animationTimelines - set LevelMenuController.status to GameData.getStatus() - call method startAnimation with random number - initialize shovel using getInstance() as it's singleton class. - use buildImage(GamePlayRoot) in Shovel class. - set GameController.data to GameData. - call SidebarElement.getSideBarElements(level, GamePlayRoot). - call gameProgress(). - if LevelMenuController.status is true: ● call method fallingSuns(rand) ● call method zombieSpawner1(rand, 25) ● call method zombieSpawner2(rand, 40) otherwise: ● create image of lawn night ● call method zombieSpawner1(rand, 25) ● call method zombieSpawner2(rand, 40)

+ void startAnimations()	- buildImage(lawn_grid) of all plants in allPlants arrayList and call attacking(GamePlayRoot). - buildImage(GamePlayRoot) of all zombies in allZombies arrayList and call moveZombie(). - set numZombiesKilled to level.getTotalZombies() multiply timeElapsed. - set progressBar using setProgress(timeElapsed).
+ void gameProgress()	- create Timeline timelineStatus with keyframe duration 1 second. Using try-catch: - set timeElapsed to numZombiesKilled divided by level.getTotalZombies(). - set progressBar using setProgress(timeElapsed). - if wonGame is -1: ● set numZombiesKilled to 0. ● call method endAnimations(). ● call method gameLost(). - else if wonGame is 0 and allZombies.size() is 0 and level.getTotalZombies() is equal to spawnedZombies: ● set numZombiesKilled to 0. ● call method endAnimations(). ● call method gameWon(). - else if progressBar.getProgress() >= 1: ● spawnZombies1.stop(). ● spawnZombies2.stop(). ● call method endAnimations(). ● call method gameWon(). - catch IOException - set timelineStatus cycleCount to INDEFINITE. - timelineStatus.play(). - add timelineStatus to animationTimelines.
+ void gameLost() throws IOException	- initialize FXMLLoader with EndGame.fxml. - initialize EndGameController. - call method endGameUI(levelNumber, false) in EndGameController - set FXMLLoader pane to GamePlayRoot.
+ void gameWon() throws IOException	- initialize FXMLLoader with EndGame.fxml. - initialize EndGameController. - call method endGameUI(levelNumber, true) in EndGameController. - set FXMLLoader pane to GamePlayRoot.
+ void endAnimations()	- Stop all Timelines in animationTimelines.
+ void updateSpawnedZombies()	- Increase spawnedZombies field by one.

+ void endZombieSpawn1()	- Stop spawnZombies1.
+ void endZombieSpawn2()	- Stop spawnZombies2.
+ void updateSunCount(int val)	- Increase sunCount by val. - call method getSunCountLabel() to setText of sunCount field.
+ void GameMenuLoader()	- Initialize new stage from GameMenu.fxml - setResizable to false. - set the scene and show the stage.
+ void removePlant(Plant plant)	- set the healthpoint of the plant to 0. - remove plant from allPlants.
+ void fallingSuns(Random rand)	- create Timeline sunDrop with keyframe duration 12 second: ● initialize Sun with random integer bound 950 as x, y is 0 and isFalling is true. ● call buildImage(GamePlayRoot) in sun class. ● call fallingSun() in sun class. - setCycleCount of sunDrop with INDEFINITE. - set sunTimeline to sunDrop. - add sunDrop to animationTimelines.
- int laneSelect(int lane)	- return y position of the lane.
+ void spawnZombies(ArrayList<Integer> zombies, int lane, int laneNumber)	- spawn zombie which type of it depends on zombies index 0.
+ void zombieSpawner1(Random rand, double time)	- create Timeline spawnZombie1 with keyframe duration is time: ● random lane number and set lane to y position. ● Using try-catch: ○ if zombieList1 first index is equal to 1: ■ call GameEntity.spawnDefaultZombie(GamePlayRoot, lane, laneNumber). ■ remove index 0 in zombieList1. ■ call method updateSpawnedZombies(). ○ else if zombieList1 first index. is equal to 2: ■ call GameEntity.FunnelHeadZombie(GamePlayRoot, lane, laneNumber). ■ remove index 0 in zombieList1.

	■ call method updateSpawnedZombies(). ○ else if zombieList1 first index is equal to 3: ■ call GameEntity.BucketHeadZombie(GamePlayRoot, lane, laneNumber). ■ remove index 0 in zombieList1. ■ call method updateSpawnedZombies(). ● catch IndexOutOfBoundsException and call method endZombieSpawn1(). - setCycleCount of spawnZombie1 with INDEFINITE. - set spawnZombies1 to spawnZombie1. - add spawnZombie1 to animationTimelines.
+ void zombieSpawner2(Random rand, double time)	- create Timeline spawnZombie2 with keyframe duration is time: ● random lane number and set lane to y position. ● Using try-catch: ○ if zombieList2 first index is equal to 1: ■ call GameEntity.spawnDefaultZombie(GamePlayRoot, lane, laneNumber). ■ remove index 0 in zombieList2. ■ call method updateSpawnedZombies(). ○ else if zombieList2 first index. is equal to 2: ■ call GameEntity.FunnelHeadZombie(GamePlayRoot, lane, laneNumber). ■ remove index 0 in zombieList2. ■ call method updateSpawnedZombies(). ○ else if zombieList2 first index is equal to 3: ■ call GameEntity.BucketHeadZombie(GamePlayRoot, lane, laneNumber). ■ remove index 0 in zombieList2.

	■ call method updateSpawnedZombies(). ● catch IndexOutOfBoundsException and call method endZombieSpawn2(). - setCycleCount of spawnZombie2 with INDEFINITE. - set spawnZombies2 to spawnZombie2. - add spawnZombie2 to animationTimelines.
+ void getGridPosition(MouseEvent event)	- get rowIndex by using lawn_grid.getRowIndex(event.getSource()) and getColumn index in the same way. - if shovel.IsDisabled() is false: ● if colIndex not null and rowIndex not null: ○ create mediaPlayer plant sound ○ iterate in allPlants and if plant.getColumn() equal to colIndex and plant.getRow() equal to rowIndex: ■ Remove plant from allPlants using removePlant method and break the loop. ● call shovel.disable(). - if SidebarElement.getCardSelected() not equal to -1: ● if colIndex not null and rowIndex not null: ○ create boolean drop = true. ○ iterate in allPlants and if plant.getColumn() equal to colIndex and plant.getRow() equal to rowIndex: ■ set drop to false and break the loop. ○ if drop is true: ■ if the cost of the card is less than or equal to sunCount. ■ call method dropPlant. ■ call method updateSunCount. ■ set card Disable on with GamePlayRoot. ○ call method .setCardSelectedToNull() in SidebarElement.
+ void dropPlant(int value, int x, int y, int row, int col)	- create plant sound. - plant to drop depends on value. ● If the value is 1, create a SunFlower. ● If the value is 2, create a PeaShooter.

	● If the value is 3, create a Wallnut. ● If the value is 4, create a CherryBomb. ● If the value is 5, create a Repeater. ● If the value is 6, create a ChilliPepper.
+ Label getSunCountLabel()	return sunCountDisplay.

6.5.2 Class GameEntity

This class deals with zombie entities and levels of the game.

6.5.2.1 Fields

Name	Description
- int level	- The game level.
- int numDefaultZombie	- The number of DefaultZombie.
- int numFunnelHeadZombie	- The number of FunnelHeadZombie.
- int numBucketHeadZombie	- The number of BucketHeadZombie.
- ArrayList<Integer> availableZombies	- Store all types of zombies.
- ArrayList<Integer> zombieList1	- Store shuffle types of zombies.
- ArrayList<Integer> zombieList2	- Store shuffle types of zombies.
- int totalZombies	- The number of total zombies.

6.5.2.2 Constructor

Name	Description
+ GameEntity(int level)	- initialize all ArrayList - in each level it has a specific number of zombies in various types. - add all types of zombies to availableZombies. - shuffle zombie in availableZombies using zombieList1 and zombieList2.

6.5.2.3 Methods

Name	Description
+ void spawnDefaultZombie(Pane pane, int lane, int laneNumber)	- create DefaultZombie with given lane and laneNumber. - buildImage using pane parameter.
+ void spawnFunnelHeadZombie(Pane pane,	- create a FunnelHeadZombie with the given

int lane, int laneNumber)	lane and laneNumber. - buildImage using pane parameter.
+ void spawnBucketHeadZombie(Pane pane, int lane, int laneNumber)	- create a BucketHeadZombie with the given lane and laneNumber. - buildImage using pane parameter.
getter of all fields	- getters.

6.5.3 Class GameData

This class has all game data and can update data when needed.

6.5.3.1 Fields

Name	Description
- List<Zombie> allZombie	all zombies in game
- List<Plant> allPlants	all plants in game
- ArrayList<Integer> zombieList1	stored shuffle types of zombie
- ArrayList<Integer> zombieList2	stored shuffle types of zombie
- final boolean status	- night mode's status is false, day mode's is true.
- final int sunCount	number of sun that player has.

6.5.3.2 Constructor

Name	Description
+ GameData(int level)	- initialize all fields.. - initialize sunCount to 50.

6.5.3.3 Methods

Name	Description
getter of all fields	getters.

6.5.4 Class EndGameController

This class deals with end game user interface.

6.5.4.1 Fields

Name	Description
- AnchorPane endGame	- main pane of this window

- ImageView zombiesAteYourBrains	- message to player
- ImageView youAteZombiesBrains	- message to player
- ImageView youWon	- message to player
- ImageView plantImage	- image of plants that players can use in the next level.
- ImageView nextLevelButton	- go to next level
+ Button mainMenuButton	- go to main menu button.

6.5.4.2 Methods

Name	Description
+ void ToMainMenu() throws IOException	- create AnchorPane with MainPage.fxml and add it to endGame.
+ void ToNextLevel() throws IOException	- create AnchorPane with LevelMenu.fxml and add it to endGame.
+ void initialize()	- set all images Visible to false except the main menu button.
+ void endGameUI(int level, boolean gameWin)	set the user interfaces when players finish game play, which are different in each level.

6.5.5 Class GameMenuController

This class deals with in-game menu user interface.

6.5.5.1 Fields

Name	Description
- Anchorpane GamePlayRoot	- main pane of this window
- ImageView restartGameButton	- restart game
- int levelNumber	- game level
- GameData data	- entity data
+ List<Plant> allPlants	- List of all plants

6.5.5.2 Methods

Name	Description
+ void initializeData(AnchorPane gamePlayRoot, int levelNumber, GameData d, List<Plant> allPlant)	- Initialize fields with given parameters.

+ void restartGame() throws IOException	- Restart game handler.
+ void showMainMenu() throws IOException	- Show Main menu handler.

6.5.6 Class MainMenuController

This class deals with the main page of the game user interface.

6.5.6.1 Fields

Name	Description
- AnchorPane mainRoot	main scene of the main page.

6.5.6.2 Methods

Name	Description
+ void exitGame()	- exit game
+ void showLevelMenu() throws Exception	- show level menu so players can pick one.

6.5.7 Class LevelMenuController

This class deals with game level menu user interface.

6.5.7.1 Fields

Name	Description
- Anchorpane levelRoot	- main pane of this window
- ImageView nightTheme	- night mode
+ ImageView dayMode	- day mode in game
+ ImageView nightMode	- night mode in game
+ boolean status = true	- night mode's status is false, day mode's is true.

6.5.8.2 Methods

Name	Description
+ void initialize()	- deal with day and night theme - lock levels that players have not reached yet.

+ void shineImage(MouseEvent event)	- make the image glow by using Glow in JavaFX.
+ void stopShining(MouseEvent event)	- stop image glow by using Glow in JavaFX.
+ void startLevel1() throws IOException	- set GameController to play game at level 1 - set it to the user interface.
+ void startLevel2() throws IOException	- set GameController to play game at level 2 - set it to the user interface.
+ void startLevel3() throws IOException	- set GameController to play game at level 3 - set it to the user interface.
+ void startLevel4() throws IOException	- set GameController to play game at level 4 - set it to the user interface.
+ void startLevel5() throws IOException	- set GameController to play game at level 5 - set it to the user interface.
+ void PrevMenuLoader() throws IOException	- set game to main page
+ void changeGameTheme()	- if it is day mode, then set to night mode otherwise, set it to day mode.
+ boolean getDayMode()	- return status.

6.6. Package application

6.6.1 class Main

This class contains the main function. Run it to start the game.

6.6.1.1 Fields

Name	Description
+ MediaPlayer mediaplayer	- Variable for sound.
- int maxLevel	- highest level that player reached.

6.6.1.2 Methods

Name	Description
+ void start(Stage primaryStage)	- Use addMusic() to play the background sound. - Initialize a new scene using the Main.fxml file. - Set the scene size to 1024x600. - setResizable to false. - Set title to “Plants VS Zombies”. - Set Scene to primaryState. - Show the primaryState

+ void addMusic()	- Initialize mediaplayer from sound. - Set mediaplayer AutoPlay to true. - Set mediaplayer CycleCount to **MediaPlayer.INDEFINITE.** - Set StartTime to 0. - Set StopTime to 50 seconds. - Play the sound using the method play().
+ void main()	- Launch the game.