VIETNAM NATIONAL UNIVERSITY - HO CHI MINH CITY

THE INTERNATIONAL UNIVERSITY

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

**OBJECT - ORIENTED PROGRAMMING**

**IT069IU**

**REPORT**

**TOPIC: Plants vs Zombie Game**

**By Group 05 – Member List**

1. Nguyễn Thế Khoa ITCSIU24042
2. Nguyễn Hưng ITCSIU24030
3. Trần Khánh Bình ITCSIU24015
4. Nguyễn Hữu Ninh ITCSIU24066
5. Hoàng Triệu Nam ITCSIU24059

**Instructor**: Dr. Trần Thanh Tùng

**Lab Instructor:** MSc. Nguyễn Trung Nghĩa

**TABLE OF CONTENT**

[**1.1 Abstract**](#_kc60kkhpuc1n) **5**

[**1.2 Application Overview**](#_aerv1jryvo7f) **5**

[**1.3 Goals**](#_pgw1ewofiz9p) **6**

[**1.4. Technique and tools used**](#_jvnjbx95cuw) **7**

[**2.1. Software requirements**](#_vbyphw68fubu) **7**

[**2.2. System Analysis**](#_9ds3sxr6unwj) **9**

[2.2.1 Diagram Design](#_p4p8938ph3gz) 9

[2.2.2 Methods](#_zg7x1xnprmpq) 10

[2.2.3 Codes](#_1iwq0zfy20qf) 11

[2.2.3.1 World](#_t3e33zgw2k7) 11

[2.2.3.1.1 Initializer](#_5gngo2tluqq) 11

[2.2.3.1.2 Logo](#_ipm4pqi0as82) 12

[2.2.3.1.3 MainMenu](#_l6e0ts81k71r) 13

[2.2.3.1.4 MyWorld](#_2mghz3c12yeg) 15

[2.2.3.1.5 Level0](#_c9uyd0v2ates) 21

[2.2.3.1.6 Level1](#_ulys6zj4iocv) 25

[2.2.3.1.7 Level2](#_svd7lonbq2h0) 30

[2.2.3.1.8 Level3](#_ava70moncskw) 35

[2.2.3.1.9 GameOver](#_jl2720d149jh) 40

[2.2.3.2 Actor](#_kku18j4q8xae) 41

[2.2.3.2.1 AHugeWave](#_m6tgieftybj9) 41

[2.2.3.2.2 DelayAudio](#_gqn456dc5bqb) 43

[2.2.3.2.3 EndTransition](#_4nsis3gybagc) 44

[2.2.3.2.4 FixOrder](#_2vzknxmdd00d) 45

[2.2.3.2.5 Grid](#_ylw2jb77ikig) 46

[2.2.3.2.6 Hitbox](#_q28od2f4rheg) 49

[2.2.3.2.7 LawnMower](#_w9cwsrz5kry2) 49

[2.2.3.2.8 ProgressionBar](#_9spzp2xmy6z4) 51

[2.2.3.2.9 SeedBank](#_7uo4gz1ti5tq) 53

[2.2.3.2.10 SeedPacket](#_tn0uwk3muhd0) 58

[2.2.3.2.10.1 SunflowerPacket](#_2d4oc0bdc3o) 62

[2.2.3.2.10.2 PeashooterPacket](#_vboq391ionnl) 63

[2.2.3.2.10.3 WalnutPacket](#_joqk34x2aiej) 64

[2.2.3.2.10.4 CactusPacket](#_qp1q6wjlxri3) 64

[2.2.3.2.10.5 RepeaterPacket](#_l8ejkf3khvqj) 65

[2.2.3.2.10.6 PotatoPacket](#_amnuefnl9g2e) 66

[2.2.3.2.11 SmoothMover](#_j6ejhhhbdpx3) 66

[2.2.3.2.11.1 Shovel](#_ujbpaer92lif) 67

[2.2.3.2.11.2 AnimatedObjects](#_ho524as3e8np) 69

[2.2.3.2.11.2.1 Buttons](#_28gagvjk91y6) 74

[2.2.3.11.2.1.1 More](#_f0z0slbuivh4) 75

[2.2.3.11.2.1.2 Retry](#_waqzavnrec1n) 77

[2.2.3.11.2.1.3 Start](#_9hspyi1gr1o7) 78

[2.2.3.2.11.2.2 Dirt](#_9fli33ct3lhh) 79

[2.2.3.2.11.2.3 Explosion](#_epaacr9uf86d) 80

[2.2.3.2.11.2.4 FallingObject](#_7uacy2mw3lg) 82

[2.2.3.2.11.2.4.1 Arm](#_xjo908wok5g4) 84

[2.2.3.2.11.2.4.2 Brick](#_7fzoe1bipv5a) 84

[2.2.3.2.11.2.4.3 Bucket](#_4okd7ytnwibe) 85

[2.2.3.2.11.2.4.4 Cone](#_yvlmgd3pw9jz) 85

[2.2.3.2.11.2.4.5 FallingSun](#_tznijbtx2q5k) 86

[2.2.3.2.11.2.4.6 Head](#_fd7jilw6ksbk) 89

[2.2.3.2.11.2.4.7 Sun](#_ow08by7l879x) 90

[2.2.3.2.11.2.4.8 WinCactus](#_t6qwwzeri51s) 94

[2.2.3.2.11.2.4.9 WinPotato](#_49tva9ai3hbp) 96

[2.2.3.2.11.2.4.10 WinRepeater](#_q316pznn0nab) 98

[2.2.3.2.11.2.5 IdleZombie](#_tgsfagi0151n) 100

[2.2.3.2.11.2.5.1 Basic](#_2962tw3eagn2) 100

[2.2.3.2.11.2.5.2 IdleBrickhead](#_hwr64m9qywz0) 101

[2.2.3.2.11.2.5.3 IdleBucket](#_8aoteb8eyswr) 102

[2.2.3.2.11.2.5.4 IdleCone](#_3oonqq7ht23t) 103

[2.2.3.2.11.2.6 Plant](#_318446mc16qz) 103

[2.2.3.2.11.2.6.1 Cactus](#_8wal0qi3xepz) 107

[2.2.3.2.11.2.6.2 Peashooter](#_z4lr5ihk7ieg) 110

[2.2.3.2.11.2.6.3 Potatomine](#_66984wzk1k8) 113

[2.2.3.2.11.2.6.4 Repeater](#_yadks49n66z4) 115

[2.2.3.2.11.2.6.5 Sunflower](#_hj7uw39zuxel) 118

[2.2.3.2.11.2.6.6 Walnut](#_8fatxow2ok49) 120

[2.2.3.2.11.2.7 Projectile](#_1flg9yiten2j) 122

[2.2.3.2.11.2.7.1 Needle](#_33sg0ues9xjl) 124

[2.2.3.2.11.2.7.2 Pea](#_9k6ctafuci3b) 125

[2.2.3.2.11.2.8 ReadySetPlant](#_x9altkpuv7y) 125

[2.2.3.2.11.2.9 Zombie](#_7a058wg1hj98) 127

[2.2.3.2.11.2.9.1 BasicZombie](#_3keqj71uatp2) 133

[2.2.3.2.11.2.9.2 Brickhead](#_zcgurgt8zcl) 136

[2.2.3.2.11.2.9.3 Buckethead](#_3mr2gd7oh8t2) 142

[2.2.3.2.11.2.9.4 Conehead](#_c3ooydoinpjt) 148

[2.2.3.2.11.2.10 fallingZombie](#_vjemm5xmgalu) 153

[2.2.3.2.11.3 useShovel](#_v0te6sjq3pb6) 154

[2.2.3.2.12 SunCounter](#_ixa6vr48eia4) 157

[2.2.3.2.13 Transistion](#_v7dfijrlupqj) 159

[2.2.3.2.14 TransparentObject](#_di5cd7ll4gbj) 162

[2.2.3.2.14.1 TransparentCactus](#_hxffhyfc1k4g) 163

[2.2.3.2.14.2 TransparentPeashooter](#_unlhjwgulyb9) 164

[2.2.3.2.14.3 TransparentPotato](#_9hus3yo4cgq6) 164

[2.2.3.2.14.4 TransparentRepeater](#_67jjzeuxjqzl) 165

[2.2.3.2.14.5 TransparentSunflower](#_jui5w2md0xqm) 166

[2.2.3.2.14.6 TransparentWallnut](#_k5jq8gbdvhwd) 166

[2.2.3.2.15 WaveManager](#_o59gfx14212z) 167

[2.2.3.2.16 finishedSending](#_quvh3cjupp0m) 174

[2.2.3.3 Extensions](#_urntfaomsmmu) 175

[2.2.3.3.1 Audio](#_n4qjmrussnhf) 175

[2.2.3.3.2 RNG](#_wmdsqkuzye5y) 177

[2.2.3.3.3 Timer1](#_rcn0m65a6b4w) 178

[**SUMMARY**](#_balu775doe3i) **179**

[3.1 Project Recap:](#_7dbqx7500b7q) 179

[3.2 Key Goals:](#_th4tm3a0no7) 180

[3.3 Methodology and Tools:](#_m33ae7kbu6u4) 180

[3.4 Outcomes and Achievements:](#_m68xbxvo6g1n) 180

[**REFERENCES**](#_rbaxgycwbx5i) **181**

INTRODUCTION

# 1.1 Abstract

Plants vs Zombie is a tower-defense game developed as an assignment for our Object-Oriented Programming (OOP) course. This project was created by a team of five first-year students from International University. Using Java and the Greenfoot platform, we have built a game where players must strategically deploy various plants to safeguard their home against a relentless invasion of zombies. Our aim is to leverage classic game mechanics with modern programming principles, enhancing our understanding of object-oriented design while delivering an engaging and thought-provoking experience. Throughout this project, we have learned the importance of teamwork, creativity, and practical coding skills, which have all contributed significantly to our growth as aspiring software developers.

# 1.2 Application Overview

– Gameplay Mechanics:

* **Plant Deployment:** Players choose plants with unique abilities and place them on a grid-based battlefield. Each plant plays a specific role—some act as barriers, others provide offensive support, all designed to counter the different types of zombies.
* **Zombie Waves:** Zombies advance from the right side of the screen toward the player's home. The game introduces progressively challenging waves, with each level featuring zombies of varying speed, strength, and special characteristics.
* **Resource Management:** Strategic planning is key, as players must manage their energy or resource points to deploy plants effectively. Resources regenerate over time, urging players to balance between offense and defense under increasing pressure.

– Key Features:

* **Interactive Interface:** Utilizing the graphical capabilities of Greenfoot, the game boasts animated sprites, engaging visual effects, and interactive user inputs that keep players immersed in the gameplay.
* **Level Design:** Currently featuring two levels, each stage offers unique layouts and obstacles. The progression in level design ensures that players continually refine their strategies to overcome heightened challenges.
* **Object-Oriented Structure:** The game is built with a clean, modular architecture. Distinct classes—such as Plant, Zombie, and World—encapsulate specific behaviors and properties, showcasing best practices in object-oriented programming and allowing for easy expansion and maintenance.
* **User Feedback and Engagement:** Real-time animations, sound effects, and score tracking provide immediate feedback, enhancing the user experience and encouraging experimentation with different defensive tactics.

– Technical Details:

* **Development Environment:** The project is developed using Java in the Greenfoot IDE, which makes learning and implementing interactive simulations both accessible and fun.
* **Code Organization:** The design follows a modular approach, with each component of the game encapsulated into its own class. This structure not only simplifies debugging and testing but also paves the way for future improvements, such as the addition of new levels or plant types.

# 1.3 Goals

– Learning Outcome:

* Enhance our understanding of Object-Oriented Programming (OOP) principles by applying them in a real-world project.

– Collaboration:

* Develop teamwork and version control skills through coordinated efforts, primarily via GitHub.

– Practical Application:

* Create a functional and engaging tower-defense game—Plants vs Zombie—that challenges players to deploy strategies and defend against waves of zombies.

– Problem-Solving:

* Design and implement robust game mechanics using Java and Greenfoot, emphasizing code modularity and reusability.

# 1.4. Technique and tools used

– Programming Language & Environment:

* **Java:** The core language used to implement the game.
* **Greenfoot:** An educational Java development environment that facilitates interactive simulations and animations.

– Design & Documentation Tools:

* **Mindmeiser:** Used to create the UML Mindmap that helped organize and conceptualize the structure of game classes and interactions. (<https://mm.tt/map/3724777316?t=MyTPSFwNut>)
* **Lucidchart:** Employed for crafting detailed UML Diagrams to accurately plan and convey the system architecture and design. (<https://lucid.app/lucidchart/56a57a19-35bf-44cc-b69f-d08fca885885/edit?viewport_loc=-4305%2C-1748%2C4445%2C1909%2CHWEp-vi-RSFO&invitationId=inv_cd6f4b55-87e1-4e41-88ae-275bc3b4083b>)

– Version Control:

* **GitHub:** Our team's repository for collaboration, code commits, and tracking project progress. (<https://github.com/trankhanhbinh/pvz-project-OOP>)

ARCHITECTING THE GAME

# 2.1. Software requirements

**Main Menu Requirements**

* **Background Music and Controls:**
  + Display the logo of the team for the start up intro
  + Automatically play looping background music using GreenfootSound when the main menu loads.
* **User Interface Layout:**
  + Display a themed main menu with a custom background image (e.g., a lawn or garden scene) and a prominent title.
  + Arrange main buttons in a fixed layout.
  + **Main Buttons:**
    - **Start:** Starts the game while stopping the menu background music and replacing it with the start game music.
    - **More:** Opens GitHub link for the project.

**Gameplay Requirements**

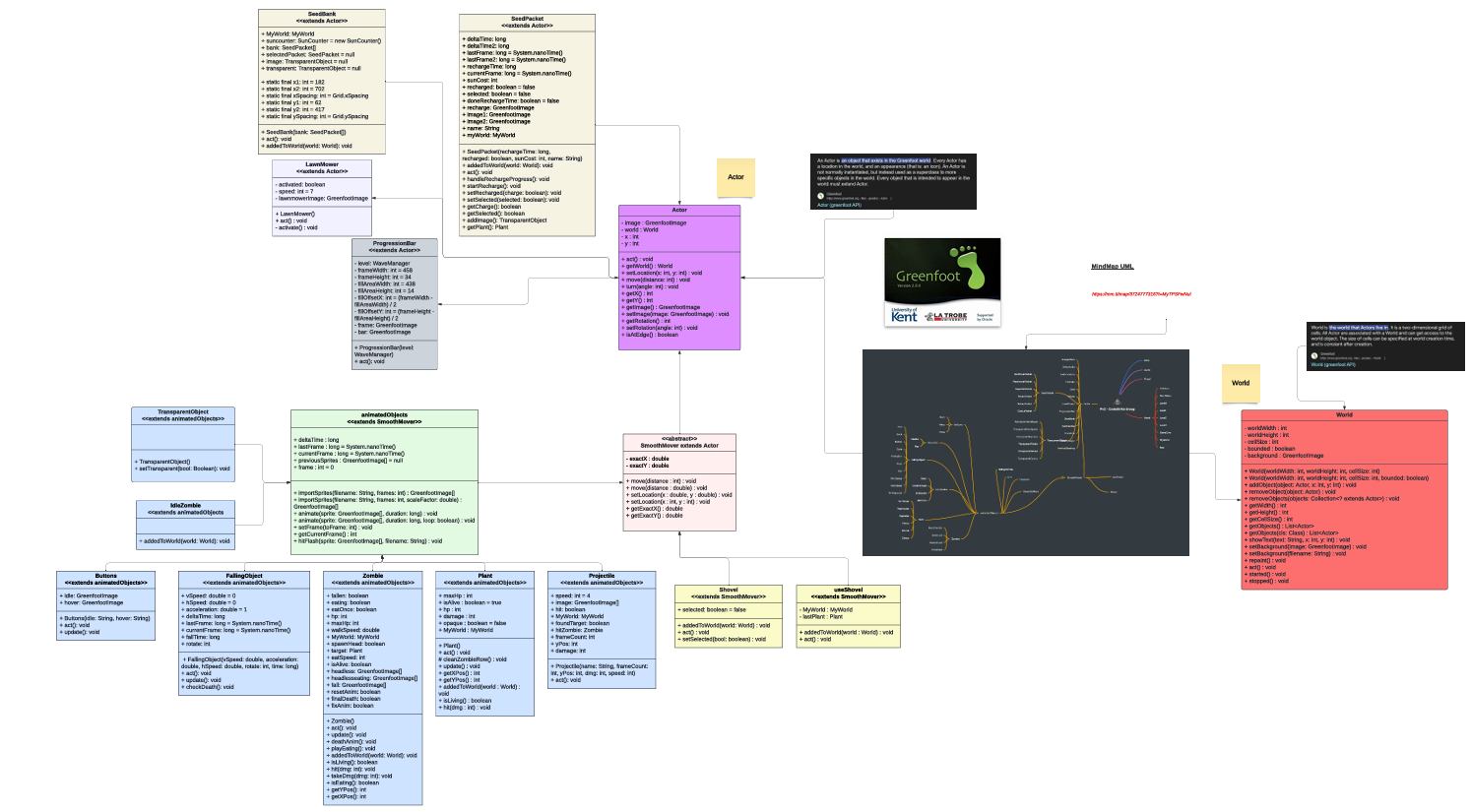
* **Game Field Setup:**
  + Display a well-defined lawn with multiple lanes where zombies will appear and lawnmover for each lane.
  + Initialize the game with a predefined level layout that determines spawn points and timings.
  + Start a cut-scene to show what kind of Zombies will appear in each level respectively.
* **Planting and Sun Points:**
  + Provide a **Seed Bank** (or plant selection bar) that lets the player choose plants to place on the lawn.
  + Deduct the appropriate sun point cost for each plant placed.
  + Periodically generate sun points by sun dropping from the sky to increase the player’s resource count.
* **Zombie Waves and Progression:**
  + Zombies should spawn in waves from the right side of the screen, with increasing difficulty as the player advances.
  + Implement a dynamic **Progression Bar** (using the techniques described in previous chapters) to show the current wave’s progress which fills from right to left.
  + Display a win animation by dropping a new plant packet (and trigger next level setup) when the level is successfully cleared.
* **Interactions and Timing:**
* Add slight delays for certain actions (e.g., when a plant shoots at zombies or when a missed shot occurs) to provide smooth visual feedback.
* Each game element’s behavior uses Greenfoot’s act() method and event handling for interactivity.

**Application Management**

* **Initialization and Execution:**
* Launch the user interface by instantiating and displaying the Main Menu (a subclass of World).
* Use Greenfoot’s threading model (if applicable) to ensure the UI updates smoothly, keeping the act() cycle as the primary game-loop driver.
* Ensure that all screen transitions (e.g., from Main Menu to Gameplay or Instructions) are managed using Greenfoot’s scene management (adding/removing objects and switching worlds).

# 2.2. System Analysis

## 2.2.1 Diagram Design

****

***Figure 2.2.1: UML Diagram***

## 2.2.2 Methods

● **Object-Oriented Programming (OOP):** The game is structured using Greenfoot's class model. Base classes (like Actor and World) are extended to create specialized classes for plants, zombies, suns, and other game elements. Each class encapsulates its own behavior and properties, enabling modular and maintainable code.

● **Event-Driven Programming:** Game interactions rely on Greenfoot’s act() method and mouse-event detection. User actions—such as selecting a seed from the Seed Bank or placing a plant on the lawn—trigger event-handlers that update game state and behavior.

● **Layout & Scene Management:** The game is divided into multiple Worlds (e.g., MainMenu, GameField, InstructionScreen) that define different scenes. Each screen arranges its components at fixed coordinates to mimic responsive layouts, ensuring consistency across different display sizes.

● **State Management with Variables:** Game progression is managed through instance variables such as currentWave, sunPoints, plantCooldown, and zombieSpawnRate. These variables track critical game states and drive dynamic responses—like triggering a new zombie wave or enabling special abilities.

● **File Persistence (Save/Load Data):** While still a work in progress, the design includes plans for saving player data (high scores, unlocked plants, and game progress) to local files. This persistence ensures that players' achievements are retained between sessions.

● **Timer-Based Delayed Execution:** Greenfoot’s time functions (using System.currentTimeMillis() alongside configurable delays) are used to schedule critical events. For example, delays control the interval between zombie spawns and the timing of plant attacks, allowing for smooth and well-paced gameplay actions.

● **Component Reusability & Navigation:** The game maximizes code reuse by leveraging common base classes. Transitions between different screens are handled by removing objects from one World and adding new ones to another, streamlining navigation and state changes.

● **Randomization:** Random elements (such as the spawning order of zombies, sun drop locations, and the shuffling of seed options) are controlled with Java’s Random utilities. This randomness injects variability into each play session, keeping gameplay fresh and unpredictable.

## **2.2.3 Codes**

### 2.2.3.1 World

#### 2.2.3.1.1 Initializer

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Initializer extends World**

**{**

**/\*\***

**\* Constructor for objects of class MyWorld.**

**\***

**\*/**

**public Initializer()**

**{**

**// Create a new world with 600x400 cells with a cell size of 1x1 pixels.**

**super(600, 448, 1, false);**

**//(width, height, each pixel is 1x1, object can appear outside the boudary)**

**setBackground(new GreenfootImage("Initializer.png"));**

**addObject(new Transition(true, new logo(), 30), 300, 224);**

**setPaintOrder(EndTransition.class, Transition.class);**

**}**

**public void act(){**

**}**

**}**

#### 2.2.3.1.2 Logo

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class logo extends World {**

**private int counter = 0;**

**private GreenfootSound menuTheme;**

**public logo(){**

**super(600, 448, 1, false);**

**setPaintOrder(EndTransition.class, Transition.class);**

**menuTheme = new GreenfootSound("menu.mp3");**

**}**

**public void act() {**

**if (!menuTheme.isPlaying()) {**

**menuTheme.setVolume(70);**

**menuTheme.playLoop();**

**}**

**counter++;**

**if (counter > 100) {**

**addObject(new Transition(true, new MainMenu(menuTheme), 6), getWidth()/2, getHeight()/2);**

**}**

**}**

**}**

#### 2.2.3.1.3 MainMenu

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class MainMenu extends World**

**{**

**Hitbox hitbox = new Hitbox();**

**GreenfootSound menutheme = new GreenfootSound("menutheme.mp3");**

**public MainMenu(GreenfootSound menutheme)**

**{**

**super(576, 430, 1, false);**

**addObject(hitbox,0,0);**

**addObject(new Start(), 428, 138);**

**addObject(new More(), 414, 233);**

**this.menutheme = menutheme;**

**Greenfoot.setSpeed(50);**

**}**

**public void act() {**

**if (Greenfoot.isKeyDown("1")) {**

**menutheme.stop();**

**Greenfoot.setWorld(new Level0());**

**} else if (Greenfoot.isKeyDown("2")) {**

**menutheme.stop();**

**Greenfoot.setWorld(new Level1());**

**} else if (Greenfoot.isKeyDown("3")) {**

**menutheme.stop();**

**Greenfoot.setWorld(new Level2());**

**} else if (Greenfoot.isKeyDown("4")) {**

**menutheme.stop();**

**Greenfoot.setWorld(new Level3());**

**}**

**}**

**public void started() {**

**if (!menutheme.isPlaying()) {**

**menutheme.setVolume(70);**

**menutheme.playLoop();**

**}**

**}**

**public void stopped() {**

**menutheme.pause();**

**}**

**public void moveHitbox() {**

**MouseInfo mouse = Greenfoot.getMouseInfo();**

**if (mouse != null) {**

**hitbox.setLocation(mouse.getX(), mouse.getY());**

**}**

**}**

**}**

#### 2.2.3.1.4 MyWorld

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**import java.util.\*;**

**public class MyWorld extends World**

**{**

**private boolean isPlaying = false;**

**public boolean lose = false;**

**public boolean loseOnce = false;**

**public boolean winOnce = false;**

**public Grid grid = new Grid();**

**public GreenfootSound Grasswalk = new GreenfootSound("Grasswalk.mp3");**

**public GreenfootSound CYS;**

**public Zombie n = null;**

**public World restartWorld;**

**public FallingObject winPlant;**

**public LawnMower lawnmower1 = new LawnMower();**

**public LawnMower lawnmower2 = new LawnMower();**

**public LawnMower lawnmower3 = new LawnMower();**

**public LawnMower lawnmower4 = new LawnMower();**

**public LawnMower lawnmower5 = new LawnMower();**

**public Zombie[][] level1 = {**

**{null, new BasicZombie(), null, null},**

**{n},**

**{new BasicZombie(), null, null, null, null},**

**{n},**

**{null, new BasicZombie(), null, new BasicZombie()},**

**{new BasicZombie()},**

**{null, null, new Conehead(), null, null},**

**{n},**

**{new BasicZombie(), new Conehead(), new BasicZombie(), new BasicZombie(), new BasicZombie(), n,new BasicZombie()},**

**{n},**

**{new Conehead(), n, null, new BasicZombie(), null, null, new BasicZombie()},**

**{new BasicZombie(),n,n, new BasicZombie(), null, new BasicZombie(), new BasicZombie()},**

**{new Buckethead(), null, null, null, null},**

**{n,new BasicZombie(),n,n,new Conehead(), n, n, new BasicZombie()},**

**{null, new BasicZombie(), null, null, new Conehead(),n,n,new BasicZombie()},**

**{new BasicZombie(), new BasicZombie(), new BasicZombie(), null, new Conehead()},**

**{null, null, new BasicZombie(), null, null},**

**{n},**

**{new Conehead(), new Conehead(), new Conehead(), new BasicZombie(), new BasicZombie(), new Buckethead(), null, new BasicZombie(), new Conehead(), new Buckethead()}**

**};**

**public SeedPacket[] bank = {new SunflowerPacket(), new PeashooterPacket(), new WalnutPacket(), new CactusPacket()};**

**public SeedBank seedbank = new SeedBank(bank);**

**public Hitbox hitbox = new Hitbox();**

**public Shovel shovel = new Shovel();**

**public WaveManager level = new WaveManager(23500L, level1, 20000L, true, 8, 18);**

**public void stopped() {**

**if (Grasswalk.isPlaying()) {**

**Grasswalk.pause();**

**}**

**}**

**public void started() {**

**if (!Grasswalk.isPlaying()) {**

**Grasswalk.playLoop();**

**}**

**Greenfoot.setSpeed(50);**

**}**

**public void moveHitbox() {**

**MouseInfo mouse = Greenfoot.getMouseInfo();**

**if (mouse != null) {**

**hitbox.setLocation(mouse.getX(), mouse.getY());**

**}**

**}**

**public void finishLevel() {**

**Grasswalk.stop();**

**Audio.play(70, "winmusic.mp3");**

**}**

**public boolean hasLost() {**

**for (Zombie i : getObjects(Zombie.class)) {**

**if (i.getWorld() != null && i.getX() < 125) {**

**lose = true;**

**return lose;**

**} else {**

**lose = false;**

**}**

**}**

**return lose;**

**}**

**public boolean hasWon() {**

**return level.hasWon();**

**}**

**public MyWorld(GreenfootSound CYS, WaveManager level, SeedBank seedbank, World restartWorld, FallingObject winPlant)**

**{**

**super(763, 448, 1, false);**

**this.CYS = CYS;**

**this.seedbank = seedbank;**

**this.restartWorld = restartWorld;**

**this.level = level;**

**this.winPlant = winPlant;**

**Greenfoot.setSpeed(50);**

**setBackground("lawn (1).png");**

**addObject(seedbank,0,0);**

**addObject(grid,0,0);**

**addObject(hitbox, 0,0);**

**addObject(shovel, 690,420);**

**addObject(new ProgressionBar(level), 490, 25);**

**addObject (lawnmower1, 132, 357);**

**addObject (lawnmower2, 132, 284);**

**addObject (lawnmower3, 132, 211);**

**addObject (lawnmower4, 132, 138);**

**addObject (lawnmower5, 132, 65);**

**setPaintOrder(ProgressionBar.class, Transition.class,AHugeWave.class, ReadySetPlant.class, SunCounter.class, useShovel.class, Shovel.class, TransparentObject.class, SeedPacket.class, FallingSun.class, Sun.class, Dirt.class, Projectile.class, FallingObject.class, Zombie.class, fallingZombie.class, Explosion.class, Plant.class);**

**}**

**public void act() {**

**if (!isPlaying) {**

**addObject(level,0,0);**

**addObject(new DelayAudio(Grasswalk,CYS, 70, true, 2000L), 0,0);**

**level.startLevel();**

**isPlaying = true;**

**}**

**if (!loseOnce && hasLost()) {**

**Grasswalk.stop();**

**Audio.play(80, "losemusic.mp3");**

**GreenfootSound scream = new GreenfootSound("scream.mp3");**

**addObject(new DelayAudio(scream, 70, false, 4000L), 0,0);**

**loseOnce = true;**

**Greenfoot.delay(250);**

**addObject(new Transition(false, new GameOver(restartWorld), "GameOver.png", 5), 365, 215);**

**} else if (!winOnce && hasWon()) {**

**winOnce = true;**

**addObject(winPlant, RNG.Int(SeedBank.x1, SeedBank.x2), 215);**

**} else {**

**if (Greenfoot.isKeyDown("1")) {**

**CYS.stop();**

**Grasswalk.stop();**

**Greenfoot.setWorld(new Level0());**

**} else if (Greenfoot.isKeyDown("2")) {**

**CYS.stop();**

**Grasswalk.stop();**

**Greenfoot.setWorld(new Level1());**

**} else if (Greenfoot.isKeyDown("3")) {**

**CYS.stop();**

**Grasswalk.stop();**

**Greenfoot.setWorld(new Level2());**

**} else if (Greenfoot.isKeyDown("4")) {**

**CYS.stop();**

**Grasswalk.stop();**

**Greenfoot.setWorld(new Level3());**

**}**

**}**

**}**

**}**

#### 2.2.3.1.5 Level0

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**import java.util.\*;**

**public class Level0 extends World**

**{**

**public GreenfootSound CYS = new GreenfootSound("chooseyourseeds.mp3");**

**public int count = 0;**

**public int scrollSpeed = 4;**

**public int location = 0;**

**public boolean started = false;**

**public Zombie n = null;**

**public SeedPacket[] bank = {new SunflowerPacket(), new PeashooterPacket(), new WalnutPacket()};**

**public SeedBank seedbank = new SeedBank(bank);**

**public Zombie[][] level1 = {**

**{new BasicZombie(), n , null, null},**

**{n},**

**{n, null, null, null, new BasicZombie()},**

**{n},**

**{null, new BasicZombie(), null, new BasicZombie()},**

**{new BasicZombie(),n,n,n,n,n,new BasicZombie()},**

**{null, null, new BasicZombie(), new BasicZombie(), new BasicZombie()},**

**{n},**

**{new BasicZombie(), new Conehead(), new BasicZombie(), new BasicZombie(), new BasicZombie(), n,new BasicZombie()}**

**};**

**public WaveManager level = new WaveManager(23500L, level1, 15000L, true, 8);**

**public Level0()**

**{**

**super(733, 430, 1, false);**

**addObject(new Basic(), 800, 200);**

**addObject(new Basic(), 900, 100);**

**addObject(new Basic(), 890, 370);**

**addObject(new Basic(), 822, 241);**

**addObject(new IdleCone(), 890, 210);**

**CYS.setVolume(70);**

**}**

**public void act() {**

**if (!started) {**

**started = true;**

**CYS.playLoop();**

**}**

**count++;**

**bgScrollAnimate();**

**}**

**public void bgScrollAnimate()**

**{**

**if (count == 100 )**

**{**

**//removeObject(message);**

**}**

**if ( count > 100 && count < 160)**

**{**

**location -= scrollSpeed;**

**scrollBGimage(location);**

**}**

**else if (count > 350 && count < 410)**

**{**

**location += scrollSpeed;**

**scrollBGimage(location);**

**}**

**else if (count == 450) {**

**List<IdleZombie> idleZombie = getObjects(IdleZombie.class );**

**for ( IdleZombie zombie : idleZombie ) {**

**removeObject(zombie);**

**}**

**}**

**else if ( count == 500 )**

**{**

**Greenfoot.setWorld(new MyWorld(CYS, level, seedbank, new Level0(), new WinPotato()));**

**}**

**}**

**public void scrollBGimage(int offset)**

**{**

**GreenfootImage bg = getBackground();**

**GreenfootImage move = new GreenfootImage("lawn (1).png");**

**bg.drawImage(move, offset, 0);**

**// get all objects and move them by the offset delta value**

**List<Actor> currentObjects = getObjects(null);**

**for ( Actor thisObject : currentObjects )**

**{**

**if ( count > 100 && count < 160)**

**{**

**thisObject.setLocation(thisObject.getX() - scrollSpeed , thisObject.getY() );**

**}**

**else if ( count > 350 && count < 410)**

**{**

**thisObject.setLocation(thisObject.getX() + scrollSpeed , thisObject.getY() );**

**}**

**}**

**}**

**}**

#### 2.2.3.1.6 Level1

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**import java.util.\*;**

**public class Level1 extends World**

**{**

**public GreenfootSound CYS = new GreenfootSound("chooseyourseeds.mp3");**

**public int count = 0;**

**public int scrollSpeed = 4;**

**public int location = 0;**

**public boolean started = false;**

**public Zombie n = null;**

**public SeedPacket[] bank = {new SunflowerPacket(), new PeashooterPacket(), new WalnutPacket(), new PotatoPacket()};**

**public SeedBank seedbank = new SeedBank(bank);**

**public Zombie[][] level1 = {**

**{null, new BasicZombie(), null, null},**

**{n},**

**{new BasicZombie(), null, null, null, null},**

**{n},**

**{null, new BasicZombie(), null, new BasicZombie()},**

**{new BasicZombie()},**

**{null, null, new Conehead(), null, null},**

**{n},**

**{new BasicZombie(), new Conehead(), new BasicZombie(), new BasicZombie(), new BasicZombie(), n,new BasicZombie()},**

**{n},**

**{new Conehead(), n, null, new BasicZombie(), null, null, new BasicZombie()},**

**{new BasicZombie(),n,n, new BasicZombie(), null, new BasicZombie(), new BasicZombie()},**

**{null, null, null, new Buckethead(), null},**

**{n,new BasicZombie(),n,n,new Conehead(), n, n, new BasicZombie()},**

**{null, new BasicZombie(), null, null, new Conehead(),n,n,new BasicZombie()},**

**{new BasicZombie(), new BasicZombie(), new BasicZombie(), null, new Conehead()},**

**{null, null, new BasicZombie(), null, null},**

**{n},**

**{new Conehead(), new Conehead(), new Conehead(), new BasicZombie(), new BasicZombie(), new Buckethead(), null, new BasicZombie(), new Conehead(), new Buckethead()}**

**};**

**public WaveManager level = new WaveManager(23500L, level1, 15000L, true, 8, 18);**

**public Level1()**

**{**

**super(733, 430, 1, false);**

**addObject(new Basic(), 800, 200);**

**addObject(new Basic(), 900, 100);**

**addObject(new Basic(), 890, 370);**

**addObject(new Basic(), 822, 241);**

**addObject(new IdleCone(), 890, 210);**

**addObject(new IdleCone(), 850, 70);**

**addObject(new IdleBucket(), 824, 317);**

**CYS.setVolume(70);**

**}**

**public void act() {**

**if (!started) {**

**started = true;**

**CYS.playLoop();**

**}**

**count++;**

**bgScrollAnimate();**

**}**

**public void bgScrollAnimate()**

**{**

**if (count == 100 )**

**{**

**//removeObject(message);**

**}**

**if ( count > 100 && count < 160)**

**{**

**location -= scrollSpeed;**

**scrollBGimage(location);**

**}**

**else if (count > 350 && count < 410)**

**{**

**location += scrollSpeed;**

**scrollBGimage(location);**

**}**

**else if (count == 450) {**

**List<IdleZombie> idleZombie = getObjects(IdleZombie.class );**

**for ( IdleZombie zombie : idleZombie ) {**

**removeObject(zombie);**

**}**

**}**

**else if ( count == 500 )**

**{**

**Greenfoot.setWorld(new MyWorld(CYS, level, seedbank, new Level1(), new WinRepeater()));**

**}**

**}**

**public void scrollBGimage(int offset)**

**{**

**GreenfootImage bg = getBackground();**

**GreenfootImage move = new GreenfootImage("lawn (1).png");**

**bg.drawImage(move, offset, 0);**

**// get all objects and move them by the offset delta value**

**List<Actor> currentObjects = getObjects(null);**

**for ( Actor thisObject : currentObjects )**

**{**

**if ( count > 100 && count < 160)**

**{**

**thisObject.setLocation(thisObject.getX() - scrollSpeed , thisObject.getY() );**

**}**

**else if ( count > 350 && count < 410)**

**{**

**thisObject.setLocation(thisObject.getX() + scrollSpeed , thisObject.getY() );**

**}**

**}**

**}**

**}**

#### 2.2.3.1.7 Level2

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**import java.util.\*;**

**/\*\***

**\* Write a description of class IntroLevel2 here.**

**\***

**\* @author (your name)**

**\* @version (a version number or a date)**

**\*/**

**public class Level2 extends World**

**{**

**public GreenfootSound CYS = new GreenfootSound("chooseyourseeds.mp3");**

**public int count = 0;**

**public int scrollSpeed = 4;**

**public int location = 0;**

**public boolean started = false;**

**public Zombie n = null;**

**public SeedPacket[] bank = {new SunflowerPacket(), new PeashooterPacket(), new WalnutPacket(), new PotatoPacket(), new RepeaterPacket()};**

**public SeedBank seedbank = new SeedBank(bank);**

**public Zombie[][] level2 = {**

**{null, n, null, new BasicZombie()},**

**{n},**

**{new BasicZombie(), null, null, null, null},**

**{n,n,new BasicZombie()},**

**{null, new BasicZombie(), new BasicZombie(), new BasicZombie()},**

**{new Conehead()},**

**{null, null, new Conehead(), null, null, n, new BasicZombie()},**

**{n},**

**{new BasicZombie(), new Buckethead(), new BasicZombie(), new Conehead(), new BasicZombie(), n,new BasicZombie()},**

**{n},**

**{new Conehead(), n, null, new BasicZombie(), null, null, new BasicZombie()},**

**{n,n,n, new Conehead(), null, new Buckethead(), new BasicZombie()},**

**{n, new Brickhead(), null, null, null},**

**{n,new BasicZombie(),new BasicZombie(), new BasicZombie(),new Conehead(), new BasicZombie(), n, new BasicZombie(), new BasicZombie()},**

**{null, new BasicZombie(), new Buckethead(), null, new Conehead(),n,n,new BasicZombie()},**

**{new BasicZombie(), new BasicZombie(), new BasicZombie(), new Brickhead(), new Conehead()},**

**{null, null, new Conehead(), null, null},**

**{n},**

**{new Brickhead(), new Conehead(), new Conehead(), new BasicZombie(), new BasicZombie(), new Buckethead(), null, new BasicZombie(), new Conehead(), new Brickhead()}**

**};**

**public WaveManager level = new WaveManager(23500L, level2, 15000L, true, 8, 18);**

**public Level2()**

**{**

**super(733, 430, 1, false);**

**addObject(new Basic(), 800, 200);**

**addObject(new IdleCone(), 900, 100);**

**addObject(new IdleCone(), 890, 370);**

**addObject(new Basic(), 822, 241);**

**addObject(new IdleBucket(), 890, 210);**

**addObject(new IdleBucket(), 850, 70);**

**addObject(new IdleBrickhead(), 824, 317);**

**CYS.setVolume(70);**

**}**

**public void act() {**

**if (!started) {**

**started = true;**

**CYS.playLoop();**

**}**

**count++;**

**bgScrollAnimate();**

**}**

**public void bgScrollAnimate()**

**{**

**if (count == 100 )**

**{**

**//removeObject(message);**

**}**

**if ( count > 100 && count < 160)**

**{**

**location -= scrollSpeed;**

**scrollBGimage(location);**

**}**

**else if (count > 350 && count < 410)**

**{**

**location += scrollSpeed;**

**scrollBGimage(location);**

**}**

**else if (count == 450) {**

**List<IdleZombie> idleZombie = getObjects(IdleZombie.class );**

**for ( IdleZombie zombie : idleZombie ) {**

**removeObject(zombie);**

**}**

**}**

**else if ( count == 500 )**

**{**

**List<IdleZombie> idleZombie = getObjects(IdleZombie.class );**

**for ( IdleZombie zombie : idleZombie ) {**

**removeObject(zombie);**

**}**

**Greenfoot.setWorld(new MyWorld(CYS,level, seedbank, new Level2(), new WinCactus()));**

**}**

**}**

**public void scrollBGimage(int offset)**

**{**

**GreenfootImage bg = getBackground();**

**GreenfootImage move = new GreenfootImage("lawn (1).png");**

**bg.drawImage(move, offset, 0);**

**// get all objects and move them by the offset delta value**

**List<Actor> currentObjects = getObjects(null);**

**for ( Actor thisObject : currentObjects )**

**{**

**if ( count > 100 && count < 160)**

**{**

**thisObject.setLocation(thisObject.getX() - scrollSpeed , thisObject.getY() );**

**}**

**else if ( count > 350 && count < 410)**

**{**

**thisObject.setLocation(thisObject.getX() + scrollSpeed , thisObject.getY() );**

**} // end inner if/else**

**} // end for-each loop**

**}**

**}**

#### 2.2.3.1.8 Level3

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**import java.util.\*;**

**public class Level3 extends World**

**{**

**public GreenfootSound CYS = new GreenfootSound("chooseyourseeds.mp3");**

**public int count = 0;**

**public int scrollSpeed = 4;**

**public int location = 0;**

**public boolean started = false;**

**public Zombie n = null;**

**public SeedPacket[] bank = {new SunflowerPacket(), new PeashooterPacket(), new WalnutPacket(), new PotatoPacket(), new RepeaterPacket(), new CactusPacket()};**

**public SeedBank seedbank = new SeedBank(bank);**

**public Zombie[][] level3 = {**

**{null, n, null, new BasicZombie()},**

**{n},**

**{new BasicZombie(), new Buckethead(), new Buckethead(),n, null},**

**{new BasicZombie(),n,new Buckethead(),new Buckethead(),n,new BasicZombie()},**

**{n, n, n, new Buckethead(),n,n,new BasicZombie(),n,n,n,new BasicZombie(),new BasicZombie()},**

**{new BasicZombie(),n,n,n,new Conehead(), n,n,n,n,n,n,n,n,n,new BasicZombie()},**

**{n},**

**{new BasicZombie(), new Conehead(), new Brickhead(), new BasicZombie(), new Conehead(), new BasicZombie(), new BasicZombie(), new Buckethead()},**

**{new BasicZombie(),n,n,new BasicZombie(),new BasicZombie()},**

**{n,n,new Buckethead(), n,n,n,n,new Buckethead()},**

**{new Conehead(), new Conehead(),new Buckethead(), new Conehead(),new Conehead()},**

**{new BasicZombie(), new Conehead(), new BasicZombie(), new BasicZombie(), new BasicZombie()},**

**{new Brickhead()},**

**{new Conehead(), new Buckethead(), new Buckethead(), new Conehead(), new Buckethead()},**

**{n},**

**{new BasicZombie(), new BasicZombie(), new BasicZombie(), new BasicZombie(), new BasicZombie()},**

**{n,n,n,new Brickhead(), new Brickhead()},**

**{new Conehead(), new Buckethead(), new Conehead()},**

**{n},**

**{new Buckethead(), new Buckethead(), new Buckethead(), new Buckethead(), new Buckethead()},**

**{new BasicZombie(), new BasicZombie(), new BasicZombie(), new BasicZombie(), new BasicZombie()},**

**{n, new Conehead()},**

**{n,n,n,n,new BasicZombie()},**

**{n},**

**{new BasicZombie(), new BasicZombie(),new BasicZombie(),new BasicZombie(),new BasicZombie()},**

**{new Conehead(),new Conehead(),new Conehead(),new Conehead(),new Conehead(), new Buckethead(),new Buckethead(),new Buckethead(),new Buckethead(),new Buckethead(), new Brickhead(),new Brickhead(),new Brickhead(),new Brickhead(),new Brickhead()},**

**{n},**

**{n,n,new Buckethead(), n,n,n,n,new Buckethead(),n,n,n,n,new Buckethead(),n,n,n,n,new Buckethead(), n,n,n,n,new Buckethead(),n,n,n,n,new Buckethead()}**

**};**

**public WaveManager level = new WaveManager(23500L, level3, 15000L, true, 7, 13, 19, 24, 27);**

**public Level3()**

**{**

**super(733, 430, 1, false);**

**addObject(new Basic(), 800, 200);**

**addObject(new Basic(), 920, 310);**

**addObject(new IdleCone(), 800, 110);**

**addObject(new IdleBucket(), 900, 100);**

**addObject(new IdleCone(), 890, 370);**

**addObject(new IdleBucket(), 822, 241);**

**addObject(new IdleBrickhead(), 890, 210);**

**addObject(new IdleBrickhead(), 850, 70);**

**addObject(new IdleBrickhead(), 824, 317);**

**CYS.setVolume(70);**

**}**

**public void act() {**

**if (!started) {**

**started = true;**

**CYS.playLoop();**

**}**

**count++;**

**bgScrollAnimate();**

**}**

**public void bgScrollAnimate()**

**{**

**if (count == 100 )**

**{**

**//removeObject(message);**

**}**

**if ( count > 100 && count < 160)**

**{**

**location -= scrollSpeed;**

**scrollBGimage(location);**

**}**

**else if (count > 350 && count < 410)**

**{**

**location += scrollSpeed;**

**scrollBGimage(location);**

**}**

**else if (count == 450) {**

**List<IdleZombie> idleZombie = getObjects(IdleZombie.class );**

**for ( IdleZombie zombie : idleZombie ) {**

**removeObject(zombie);**

**}**

**}**

**else if ( count == 500 )**

**{**

**List<IdleZombie> idleZombie = getObjects(IdleZombie.class );**

**for ( IdleZombie zombie : idleZombie ) {**

**removeObject(zombie);**

**}**

**Greenfoot.setWorld(new MyWorld(CYS,level, seedbank, new Level3(), new WinPotato()));**

**}**

**}**

**public void scrollBGimage(int offset)**

**{**

**GreenfootImage bg = getBackground();**

**GreenfootImage move = new GreenfootImage("lawn (1).png");**

**bg.drawImage(move, offset, 0);**

**// get all objects and move them by the offset delta value**

**List<Actor> currentObjects = getObjects(null);**

**for ( Actor thisObject : currentObjects )**

**{**

**if ( count > 100 && count < 160)**

**{**

**thisObject.setLocation(thisObject.getX() - scrollSpeed , thisObject.getY() );**

**}**

**else if ( count > 350 && count < 410)**

**{**

**thisObject.setLocation(thisObject.getX() + scrollSpeed , thisObject.getY() );**

**} // end inner if/else**

**} // end for-each loop**

**}**

**}**

#### 2.2.3.1.9 GameOver

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class GameOver extends World {**

**// Private instance variable to track the mouse hitbox.**

**public Hitbox hitbox;**

**/\*\***

**\* Constructor for GameOver screen.**

**\* @param restart The world to restart when the player chooses to retry.**

**\*/**

**public GameOver(World restart) {**

**super(763, 448, 1, false);**

**hitbox = new Hitbox();**

**addObject(hitbox, 0, 0);**

**Greenfoot.setSpeed(50);**

**addObject(new Retry(restart), 380, 405);**

**}**

**public void act() {**

**moveHitbox();**

**}**

**public void moveHitbox() {**

**MouseInfo mouse = Greenfoot.getMouseInfo();**

**if (mouse != null) {**

**hitbox.setLocation(mouse.getX(), mouse.getY());**

**}**

**}**

**}**

### 2.2.3.2 Actor

#### 2.2.3.2.1 AHugeWave

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class AHugeWave extends Actor{**

**public int counter = 0;**

**public boolean finalWave = false;**

**public AHugeWave(boolean finalWave) {**

**this.finalWave = finalWave;**

**}**

**public void act()**

**{**

**counter++;**

**if (!finalWave) {**

**if (counter > 300) {**

**Audio.play(70, "siren.mp3");**

**getWorld().removeObject(this);**

**return;**

**}**

**} else {**

**if (counter == 300) {**

**Audio.play(70, "siren.mp3");**

**setImage("finalwave.png");**

**Audio.play(70, "finalwave.mp3");**

**}**

**if (counter > 450) {**

**getWorld().removeObject(this);**

**return;**

**}**

**}**

**}**

**}**

#### 2.2.3.2.2 DelayAudio

**import greenfoot.\*;**

**class DelayAudio extends Actor**

**{**

**public GreenfootSound music;**

**public GreenfootSound stop;**

**public int volume;**

**public boolean loop;**

**public long deltaTime;**

**public long lastFrame = System.nanoTime();**

**public long currentFrame = System.nanoTime();**

**public long delayTime;**

**DelayAudio(GreenfootSound music, int volume, boolean loop, long delayTime){**

**this.delayTime = delayTime;**

**this.music = music;**

**this.volume = volume;**

**this.loop = loop;**

**}**

**DelayAudio(GreenfootSound music, GreenfootSound stop, int volume, boolean loop, long delayTime){**

**this.stop = stop;**

**this.delayTime = delayTime;**

**this.music = music;**

**this.volume = volume;**

**this.loop = loop;**

**}**

**public void act() {**

**currentFrame = System.nanoTime();**

**deltaTime = (currentFrame - lastFrame)/1000000;**

**if (deltaTime > delayTime){**

**if (stop != null) {**

**stop.stop();**

**}**

**music.setVolume(volume);**

**if (loop) {**

**music.playLoop();**

**} else {**

**music.play();**

**}**

**getWorld().removeObject(this);**

**return;**

**}**

**}**

**}**

#### 2.2.3.2.3 EndTransition

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class EndTransition extends Actor {**

**private int a = 255;**

**/\*\***

**\* Constructor: Initializes the EndTransition actor by setting its image transparency to fully opaque.**

**\*/**

**public EndTransition() {**

**getImage().setTransparency(a);**

**}**

**/\*\***

**\* Act method: Called each act cycle.**

**\* Reduces the image transparency by 10 units at a time. When fully faded out,**

**\* it removes the actor from the world.**

**\*/**

**public void act() {**

**a -= 10;**

**if (a >= 0) {**

**getImage().setTransparency(a);**

**} else {**

**getImage().setTransparency(0);**

**getWorld().removeObject(this);**

**}**

**}**

**}**

#### 2.2.3.2.4 FixOrder

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**class FixOrder extends Actor {**

**private long startTime;**

**private long delayTime;**

**private WaveManager level;**

**public FixOrder(WaveManager level, long delayTime) {**

**this.level = level;**

**this.delayTime = delayTime;**

**startTime = System.currentTimeMillis(); // Store initial time**

**}**

**public void act() {**

**long elapsedTime = System.currentTimeMillis() - startTime;**

**if (elapsedTime > delayTime) {**

**level.fixOrder(); // Execute fixOrder after delay**

**getWorld().removeObject(this); // Remove once done**

**}**

**}**

**}**

#### 2.2.3.2.5 Grid

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Grid extends Actor{**

**// instance variables - replace the example below with your own**

**public Plant[][] Grid = new Plant[5][9];**

**public static final int xOffset = 221;**

**public static final int yOffset = 99;**

**public static final int xSpacing = 60;**

**public static final int ySpacing = 75;**

**/\*\***

**\* Constructor for objects of class Grid**

**\*/**

**public Grid()**

**{**

**}**

**public void placePlant(int x, int y, Plant plant) {**

**if (Grid[y][x] == null) {**

**Grid[y][x] = plant;**

**World MyWorld = getWorld();**

**MyWorld.addObject(plant, x\*xSpacing+xOffset, y\*ySpacing+yOffset);**

**Audio.play(80, "plant.mp3", "plant2.mp3");**

**}**

**}**

**public Plant getPlant(int x, int y) {**

**return Grid[y][x];**

**}**

**public void removePlant(int x, int y) {**

**if (Grid[y][x] != null) {**

**getWorld().removeObject(Grid[y][x]);**

**Grid[y][x] = null;**

**}**

**Audio.play(80,"plant2.mp3");**

**}**

**public void updateGrid() {**

**for (int i = 0; i < Grid.length; i++) {**

**for (int k = 0; k < Grid[0].length; k++) {**

**if (Grid[i][k] != null) {**

**World MyWorld = getWorld();**

**Plant temp = Grid[i][k];**

**MyWorld.addObject(temp, k\*xSpacing+xOffset, i\*ySpacing+yOffset);**

**}**

**}**

**}**

**}**

**/\*\***

**\* Act - do whatever the Grid wants to do. This method is called whenever**

**\* the 'Act' or 'Run' button gets pressed in the environment.**

**\*/**

**public void act()**

**{**

**}**

**}**

#### 2.2.3.2.6 Hitbox

**import greenfoot.\*;**

**import java.util.\*;**

**public class Hitbox extends Actor{**

**public void act(){**

**}**

**public List getTouching(){**

**return getIntersectingObjects(null);**

**}**

**}**

#### 2.2.3.2.7 LawnMower

**import greenfoot.\*;**

**import java.util.\*;**

**public class LawnMower extends Actor {**

**private boolean activated;**

**private int speed = 7;**

**private GreenfootImage lawnmowerImage;**

**public LawnMower() {**

**activated = false;**

**lawnmowerImage = new GreenfootImage("lawn\_mower.gif");**

**setImage(lawnmowerImage);**

**}**

**public void act() {**

**if (getWorld() == null) {**

**return;**

**}**

**if (!activated) {**

**List<Zombie> zombies = new ArrayList<>(getWorld().getObjects(Zombie.class));**

**for (Zombie z : zombies) {**

**if (Math.abs(z.getY() - getY()) < getImage().getHeight() / 4 && z.getX() < getX() + 100) {**

**activate();**

**break;**

**}**

**}**

**} else {**

**setLocation(getX() + speed, getY());**

**int removalTolerance = 20;**

**List<Zombie> intersectingZombies = new ArrayList<>(getIntersectingObjects(Zombie.class));**

**for (Zombie z : intersectingZombies) {**

**try {**

**if (z != null && z.getWorld() != null && Math.abs(z.getY() - getY()) <= removalTolerance) {**

**getWorld().removeObject(z);**

**}**

**} catch (IllegalStateException ise) {**

**}**

**}**

**if (getX() >= getWorld().getWidth() - 4) {**

**getWorld().removeObject(this);**

**}**

**}**

**}**

**private void activate() {**

**activated = true;**

**}**

**}**

#### 2.2.3.2.8 ProgressionBar

**import greenfoot.\*;**

**import greenfoot.Color;**

**public class ProgressionBar extends Actor {**

**private WaveManager level;**

**private int frameWidth = 458;**

**private int frameHeight = 34;**

**private int fillAreaWidth = 438;**

**private int fillAreaHeight = 14;**

**private int fillOffsetX = (frameWidth - fillAreaWidth) / 2;**

**private int fillOffsetY = (frameHeight - fillAreaHeight) / 2;**

**private GreenfootImage frame;**

**private GreenfootImage bar;**

**public ProgressionBar(WaveManager level) {**

**this.level = level;**

**frame = new GreenfootImage("progress.png");**

**if(frame == null) {**

**System.err.println("Error: progress.png could not be loaded. Check your images folder.");**

**frame = new GreenfootImage(frameWidth, frameHeight);**

**frame.setColor(Color.BLACK);**

**frame.fill();**

**}**

**frame.scale(frameWidth, frameHeight);**

**bar = new GreenfootImage(frame);**

**setImage(bar);**

**}**

**public void act() {**

**if (level.wave != -1) {**

**int currentFillWidth = (int)(fillAreaWidth \* ((double)(level.wave + 1) / level.level.length));**

**bar.drawImage(frame, 0, 0);**

**bar.setColor(new Color(80, 160, 32));**

**bar.fillRect(fillOffsetX + fillAreaWidth - currentFillWidth, fillOffsetY, currentFillWidth, fillAreaHeight);**

**setImage(bar);**

**}**

**}**

**}**

#### 2.2.3.2.9 SeedBank

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**import java.util.\*;**

**public class SeedBank extends Actor**

**{**

**/\*\***

**\* Act - do whatever the SeedBank wants to do. This method is called whenever**

**\* the 'Act' or 'Run' button gets pressed in the environment.**

**\*/**

**public MyWorld MyWorld;**

**public SunCounter suncounter = new SunCounter();**

**public SeedPacket[] bank;**

**public SeedPacket selectedPacket = null;**

**public TransparentObject image = null;**

**public TransparentObject transparent = null;**

**public static final int x1 = 182;**

**public static final int x2 = 702;**

**public static final int xSpacing = Grid.xSpacing;**

**public static final int y1 = 62;**

**public static final int y2 = 417;**

**public static final int ySpacing = Grid.ySpacing;**

**public SeedBank(SeedPacket[] bank) {**

**this.bank = bank;**

**}**

**public void act() {**

**MouseInfo mouse = Greenfoot.getMouseInfo();**

**if (mouse != null) {**

**if (image != null) {**

**if ((mouse.getX() < x1 || mouse.getX() > x2 || mouse.getY() < y1 || mouse.getY() > y2)**

**|| (MyWorld.grid.Grid[(int)((mouse.getY()-y1)/ySpacing)][(int)((mouse.getX()-x1)/xSpacing)] != null)) {**

**image.setTransparent(false);**

**image.setLocation(mouse.getX(), mouse.getY());**

**} else {**

**int x = (int)((mouse.getX()-x1)/xSpacing);**

**int y = (int)((mouse.getY()-y1)/ySpacing);**

**image.setTransparent(true);**

**image.setLocation(x\*Grid.xSpacing+Grid.xOffset, y\*Grid.ySpacing+Grid.yOffset);**

**}**

**} else {**

**}**

**if (Greenfoot.mouseClicked(null)) {**

**MyWorld.moveHitbox();**

**//Debug: System.out.println(mouse.getX()+" "+ mouse.getY());**

**if (image != null) {**

**if (mouse.getX() < x1 || mouse.getX() > x2 || mouse.getY() < y1 || mouse.getY() > y2) {**

**MyWorld.removeObject(image);**

**image = null;**

**boolean selected = false;**

**Audio.play(80,"tap.mp3", "tap2.mp3");**

**if (MyWorld.hitbox.getTouching().contains(selectedPacket)) {**

**selected = true;**

**}**

**if (!selected) {**

**selectedPacket.setSelected(false);**

**selectedPacket = null;**

**}**

**} else {**

**int x = (int)((mouse.getX()-x1)/xSpacing);**

**int y = (int)((mouse.getY()-y1)/ySpacing);**

**if (MyWorld.grid.Grid[y][x] == null) {**

**MyWorld.grid.placePlant(x, y, selectedPacket.getPlant());**

**suncounter.removeSun(selectedPacket.sunCost);**

**getWorld().removeObject(image);**

**image = null;**

**selectedPacket.startRecharge();**

**selectedPacket.setRecharged(false);**

**selectedPacket.setSelected(false);**

**selectedPacket = null;**

**}**

**}**

**}**

**for (Object i : MyWorld.hitbox.getTouching()) {**

**if (i instanceof SeedPacket) {**

**SeedPacket clicked = (SeedPacket)i;**

**if (selectedPacket != clicked) {**

**if (clicked.recharged) {**

**if (selectedPacket != null) {**

**selectedPacket.setSelected(false);**

**selectedPacket = null;**

**}**

**selectedPacket = clicked;**

**clicked.setSelected(true);**

**Audio.play(80, "seedlift.mp3");**

**image = clicked.addImage();**

**} else {**

**Audio.play(80, "buzzer.mp3");**

**}**

**} else {**

**if (clicked.recharged) {**

**selectedPacket = null;**

**clicked.setSelected(false);**

**Audio.play(80, "seedlift.mp3");**

**}**

**}**

**}**

**}**

**}**

**}**

**}**

**@Override**

**public void addedToWorld(World world) {**

**MyWorld = (MyWorld)getWorld();**

**MyWorld.addObject(suncounter, 120, 50);**

**for (int i = 0; i < bank.length; i++) {**

**MyWorld.addObject(bank[i], 120, 120+i\*50);**

**}**

**}**

**}**

#### 2.2.3.2.10 SeedPacket

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**/\*\***

**\* Write a description of class SeedPacket here.**

**\***

**\* @author (your name)**

**\* @version (a version number or a date)**

**\*/**

**public class SeedPacket extends Actor {**

**public long deltaTime;**

**public long deltaTime2;**

**public long lastFrame = System.nanoTime();**

**public long lastFrame2 = System.nanoTime();**

**public long rechargeTime;**

**public long currentFrame = System.nanoTime();**

**public int sunCost;**

**public boolean recharged = false;**

**public boolean selected = false;**

**public boolean doneRechargeTime = false;**

**public GreenfootImage recharge;**

**public GreenfootImage image1;**

**public GreenfootImage image2;**

**public String name;**

**public MyWorld myWorld;**

**public SeedPacket(long rechargeTime, boolean recharged, int sunCost, String name){**

**this.rechargeTime = rechargeTime;**

**this.recharged = recharged;**

**this.sunCost = sunCost;**

**this.name = name;**

**this.image1 = new GreenfootImage(name + "1.png");**

**this.image2 = new GreenfootImage(name + "2.png");**

**}**

**//create or start, I forgot what I called this similar method in the previous class I made**

**@Override**

**public void addedToWorld(World world){**

**setRecharged(recharged);**

**doneRechargeTime = recharged;**

**setImage(recharged ? image1 : image2);**

**myWorld = (MyWorld) getWorld();**

**recharge = new GreenfootImage(getImage().getWidth(), getImage().getHeight());**

**}**

**// check if the valid condition (Sun and Time cooldown) is met**

**@Override**

**public void act() {**

**currentFrame = System.nanoTime();**

**deltaTime = (currentFrame - lastFrame)/1000000;**

**deltaTime2 = (currentFrame - lastFrame2)/1000000;**

**handleRechargeProgress();**

**if (myWorld.seedbank.suncounter.sun >= sunCost){**

**setRecharged(deltaTime > rechargeTime);**

**} else {**

**setRecharged(false);**

**}**

**}**

**//animation+timing**

**public void handleRechargeProgress() {**

**if (deltaTime < rechargeTime && !doneRechargeTime) {**

**if (!recharged && deltaTime2 > 500L) {**

**setImage(name + "2.png");**

**recharge.clear();**

**recharge.setColor(Color.BLACK);**

**int height = getImage().getHeight() - (int)Math.round(getImage().getHeight() \* ((double) deltaTime/rechargeTime));**

**recharge.setTransparency(110);**

**recharge.fillRect(0, 0, getImage().getWidth(), height);**

**getImage().drawImage(recharge, 0, 0);**

**lastFrame2 = System.nanoTime();**

**}**

**} else if (!recharged && !doneRechargeTime) {**

**doneRechargeTime = true;**

**setImage(image2);**

**}**

**}**

**//start recharge**

**public void startRecharge(){**

**lastFrame = currentFrame;**

**doneRechargeTime = false;**

**}**

**public void setRecharged(boolean charge){**

**if (recharged != charge){**

**recharged = charge;**

**setImage(recharged ? image1 : image2);**

**}**

**}**

**//It's 11pm rn I need to sleep**

**public void setSelected(boolean selected){**

**this.selected = selected;**

**setImage(selected ? image2 : image1);**

**}**

**public boolean getCharge(){**

**return recharged;**

**}**

**public boolean getSelected(){**

**return selected;**

**}**

**public TransparentObject addImage(){**

**return null;**

**}**

**public Plant getPlant(){**

**return null;**

**}**

**}**

##### 2.2.3.2.10.1 SunflowerPacket

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class SunflowerPacket extends SeedPacket**

**{**

**/\*\***

**\* Act - do whatever the SunflowerPacket wants to do. This method is called whenever**

**\* the 'Act' or 'Run' button gets pressed in the environment.**

**\*/**

**public SunflowerPacket() {**

**super(8000L, true,50, "sunflowerpacket");**

**}**

**public TransparentObject addImage() {**

**TransparentObject temp = new TransparentSunflower(false);**

**((MyWorld)getWorld()).addObject(temp, Greenfoot.getMouseInfo().getX(), Greenfoot.getMouseInfo().getY());**

**return temp;**

**}**

**public Plant getPlant() {**

**return new Sunflower();**

**}**

**}**

##### 2.2.3.2.10.2 PeashooterPacket

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class PeashooterPacket extends SeedPacket{**

**public PeashooterPacket() {**

**super(10000L, true, 100, "peashooterpacket");**

**}**

**public TransparentObject addImage() {**

**TransparentObject temp = new TransparentPeashooter(false);**

**((MyWorld)getWorld()).addObject(temp, Greenfoot.getMouseInfo().getX(), Greenfoot.getMouseInfo().getY());**

**return temp;**

**}**

**public Plant getPlant() {**

**return new Peashooter();**

**}**

**}**

##### 2.2.3.2.10.3 WalnutPacket

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class WalnutPacket extends SeedPacket**

**{**

**/\*\***

**\* Act - do whatever the WalnutPacket wants to do. This method is called whenever**

**\* the 'Act' or 'Run' button gets pressed in the environment.**

**\*/**

**public WalnutPacket() {**

**super(40000L, false, 50, "walnutpacket");**

**}**

**public TransparentObject addImage() {**

**TransparentObject temp = new TransparentWalnut(false);**

**((MyWorld)getWorld()).addObject(temp, Greenfoot.getMouseInfo().getX(), Greenfoot.getMouseInfo().getY());**

**return temp;**

**}**

**public Plant getPlant() {**

**return new Walnut();**

**}**

**}**

##### 2.2.3.2.10.4 CactusPacket

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class CactusPacket extends SeedPacket**

**{**

**public CactusPacket() {**

**super(20000L, false, 125, "cactuspacket");**

**}**

**public TransparentObject addImage() {**

**TransparentObject temp = new TransparentCactus(false);**

**((MyWorld)getWorld()).addObject(temp, Greenfoot.getMouseInfo().getX(), Greenfoot.getMouseInfo().getY());**

**return temp;**

**}**

**public Plant getPlant() {**

**return new Cactus();**

**}**

**}**

##### 2.2.3.2.10.5 RepeaterPacket

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class RepeaterPacket extends SeedPacket**

**{**

**public RepeaterPacket() {**

**super(11000L, true, 225, "repeaterpacket");**

**}**

**public TransparentObject addImage() {**

**TransparentObject temp = new TransparentRepeater(false);**

**((MyWorld)getWorld()).addObject(temp, Greenfoot.getMouseInfo().getX(), Greenfoot.getMouseInfo().getY());**

**return temp;**

**}**

**public Plant getPlant() {**

**return new Repeater();**

**}**

**}**

##### 2.2.3.2.10.6 PotatoPacket

#### 2.2.3.2.11 SmoothMover

**import greenfoot.\*; // (World, Actor, GreenfootImage, and Greenfoot)**

**public abstract class SmoothMover extends Actor{**

**private double exactX;**

**private double exactY;**

**@Override**

**public void move(int distance){**

**move((double) distance);**

**}**

**public void move(double distance){**

**double radians = Math.toRadians(getRotation());**

**double dx = Math.cos(radians) \* distance;**

**double dy = Math.sin(radians) \* distance;**

**setLocation(exactX + dx, exactY + dy);**

**}**

**public void setLocation(double x, double y){**

**exactX = x;**

**exactY = y;**

**super.setLocation((int) Math.round(x), (int) Math.round(y));**

**}**

**@Override**

**public void setLocation(int x, int y){**

**exactX = x;**

**exactY = y;**

**super.setLocation(x, y);**

**}**

**public double getExactX(){**

**return exactX;**

**}**

**public double getExactY(){**

**return exactY;**

**}**

**}**

##### 2.2.3.2.11.1 Shovel

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**/\*\***

**\* Write a description of class Shovel here.**

**\***

**\* @author (your name)**

**\* @version (a version number or a date)**

**\*/**

**public class Shovel extends SmoothMover**

**{**

**/\*\***

**\* Act - do whatever the Shovel wants to do. This method is called whenever**

**\* the 'Act' or 'Run' button gets pressed in the environment.**

**\*/**

**public boolean selected = false;**

**public void addedToWorld(World world) {**

**setImage("shovel1.png");**

**selected = false;**

**}**

**public void act()**

**{**

**MouseInfo mouse = Greenfoot.getMouseInfo();**

**if (mouse != null) {**

**if (Greenfoot.mouseClicked(null)) {**

**MyWorld MyWorld = (MyWorld)getWorld();**

**MyWorld.moveHitbox();**

**if (intersects(MyWorld.hitbox)) {**

**if (!selected) {**

**selected = true;**

**setImage("shovel2.png");**

**Audio.play(80, "shovel.mp3");**

**MyWorld.addObject(new useShovel(), mouse.getX(), mouse.getY());**

**}**

**}**

**}**

**}**

**}**

**public void setSelected(boolean bool) {**

**if (!bool) {**

**selected = bool;**

**setImage("shovel1.png");**

**}**

**}**

**}**

##### 2.2.3.2.11.2 AnimatedObjects

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**import java.util.Timer;**

**import java.io.\*;**

**public class animatedObjects extends SmoothMover{**

**/\*\***

**\* Act - do whatever the animatedObjects wants to do. This method is called whenever**

**\* the 'Act' or 'Run' button gets pressed in the environment.**

**\*/**

**public long deltaTime;**

**public long lastFrame = System.nanoTime();**

**public long currentFrame = System.nanoTime();**

**public GreenfootImage[] previousSprites = null;**

**public int frame = 0;**

**//In milliseconds (ms)**

**public GreenfootImage[] importSprites(String filename, int frames) {**

**GreenfootImage[] temp = new GreenfootImage[frames];**

**for (int i = 0; i < frames; i++) {**

**GreenfootImage scale = new GreenfootImage(filename+(i+1)+".png");**

**temp[i] = scale;**

**}**

**return temp;**

**}**

**public GreenfootImage[] importSprites(String filename, int frames, double scaleFactor) {**

**GreenfootImage[] temp = new GreenfootImage[frames];**

**for (int i = 0; i < frames; i++) {**

**GreenfootImage scale = new GreenfootImage(filename+(i+1)+".png");**

**scale.scale((int)(scale.getWidth()\*scaleFactor), (int)(scale.getHeight()\*scaleFactor));**

**temp[i] = scale;**

**}**

**return temp;**

**}**

**public void animate(GreenfootImage[] sprite, long duration) {**

**currentFrame = System.nanoTime();**

**deltaTime = (currentFrame - lastFrame) / 1000000;**

**if (deltaTime > duration) {**

**lastFrame = currentFrame;**

**frame++;**

**if (frame < sprite.length) {**

**setImage(sprite[frame]);**

**}**

**}**

**if (frame < sprite.length) {**

**} else {**

**frame = 0;**

**setImage(sprite[frame]);**

**}**

**if (!sprite.equals(previousSprites)) {**

**setImage(sprite[frame]);**

**previousSprites = sprite;**

**}**

**}**

**public void animate(GreenfootImage[] sprite, long duration, boolean loop) {**

**currentFrame = System.nanoTime();**

**deltaTime = (currentFrame - lastFrame) / 1000000;**

**if (deltaTime > duration) {**

**lastFrame = currentFrame;**

**frame++;**

**if (frame < sprite.length) {**

**setImage(sprite[frame]);**

**}**

**}**

**if (frame < sprite.length) {**

**} else if (loop) {**

**frame = 0;**

**setImage(sprite[frame]);**

**} else {**

**//do nothing**

**}**

**if (!sprite.equals(previousSprites)) {**

**if (frame < sprite.length) {**

**setImage(sprite[frame]);**

**previousSprites = sprite;**

**}**

**}**

**}**

**public void setFrame(int toFrame) {**

**frame = toFrame-1;**

**}**

**public int getCurrentFrame() {**

**return frame+1;**

**}**

**public void hitFlash(GreenfootImage[] sprite, String filename) {**

**int tempFrame = frame;**

**int flashFrame = tempFrame+1;**

**if (tempFrame >= sprite.length) {**

**tempFrame = 0;**

**flashFrame = 1;**

**} else if (flashFrame >= sprite.length) {**

**flashFrame = 0;**

**}**

**GreenfootImage first = new GreenfootImage("flash"+filename+(tempFrame+1)+".png");**

**GreenfootImage second = new GreenfootImage("flash"+filename+(flashFrame+1)+".png");**

**sprite[tempFrame] = first;**

**sprite[flashFrame] = second;**

**setImage(sprite[tempFrame]);**

**Timer timer = new Timer();**

**timer.schedule(new Timer1(sprite, filename, tempFrame,flashFrame), 500L);**

**}**

**}**

###### 2.2.3.2.11.2.1 Buttons

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Buttons extends animatedObjects{**

**public GreenfootImage idle;**

**public GreenfootImage hover;**

**public Buttons(String idle, String hover) {**

**this.idle = new GreenfootImage(idle);**

**this.hover = new GreenfootImage(hover);**

**setImage(idle);**

**}**

**public void act()**

**{**

**MouseInfo mouse = Greenfoot.getMouseInfo();**

**GameOver world = (GameOver)getWorld();**

**if (mouse != null) {**

**world.moveHitbox();**

**if (this.intersects(world.hitbox)) {**

**setImage(hover);**

**} else {**

**setImage(idle);**

**}**

**if (Greenfoot.mouseClicked(this)) {**

**Audio.play(100, "gravebutton.mp3");**

**update();**

**}**

**}**

**}**

**public void update() {**

**}**

**}**

###### 2.2.3.11.2.1.1 More

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**import java.awt.Desktop;**

**import java.net.URL;**

**public class More extends Buttons**

**{**

**public boolean clicked = false;**

**GreenfootImage[] start;**

**public int counter = 0;**

**public More() {**

**super("more1.png", "more2.png");**

**start = importSprites("more", 2);**

**}**

**public void act()**

**{**

**MouseInfo mouse = Greenfoot.getMouseInfo();**

**MainMenu world = (MainMenu)getWorld();**

**if (mouse != null) {**

**world.moveHitbox();**

**if (this.intersects(world.hitbox)) {**

**setImage(hover);**

**} else {**

**setImage(idle);**

**}**

**if (Greenfoot.mouseClicked(this)) {**

**clicked = true;**

**try{**

**Desktop.getDesktop().browse(new URL("https://github.com/trankhanhbinh/pvz-project-OOP").toURI());**

**}**

**catch (Exception e)**

**{}**

**Audio.play(100, "gravebutton.mp3");**

**}**

**}**

**}**

**public void update() {**

**}**

**}**

###### 2.2.3.11.2.1.2 Retry

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Retry extends Buttons**

**{**

**public World restart;**

**public Retry(World restart) {**

**super("retry1.png", "retry2.png");**

**this.restart = restart;**

**}**

**public void update() {**

**getWorld().addObject(new Transition(false, restart, 10), 365, 215);**

**}**

**}**

###### 2.2.3.11.2.1.3 Start

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Start extends Buttons**

**{**

**public boolean clicked = false;**

**GreenfootImage[] start;**

**public int counter = 0;**

**public Start() {**

**super("start1.png", "start2.png");**

**start = importSprites("start", 2);**

**}**

**public void act()**

**{**

**MouseInfo mouse = Greenfoot.getMouseInfo();**

**MainMenu world = (MainMenu)getWorld();**

**if (clicked) {**

**animate(start, 80, true);**

**counter++;**

**if (counter == 200) {**

**update();**

**}**

**} else {**

**if (mouse != null) {**

**world.moveHitbox();**

**if (this.intersects(world.hitbox)) {**

**setImage(hover);**

**} else {**

**setImage(idle);**

**}**

**if (Greenfoot.mouseClicked(this)) {**

**clicked = true;**

**world.menutheme.stop();**

**Audio.play(100, "gravebutton.mp3");**

**Audio.play(70, "losemusic.mp3");**

**getWorld().addObject(new DelayAudio(new GreenfootSound("evillaugh.mp3"), 70, false, 1000L), 0,0);**

**}**

**}**

**}**

**}**

**public void update() {**

**getWorld().addObject(new Transition(false, new Level0(), 4), 381, 224);**

**}**

**}**

###### 2.2.3.2.11.2.2 Dirt

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Dirt extends animatedObjects**

**{**

**public GreenfootImage[] dirt;**

**public Dirt() {**

**dirt = importSprites("dirt",4);**

**}**

**public void act()**

**{**

**if (frame <= 3) {**

**animate(dirt, 50L, false);**

**} else {**

**getWorld().removeObject(this);**

**return;**

**}**

**}**

**}**

###### 2.2.3.2.11.2.3 Explosion

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**import java.util.ArrayList;**

**public class Explosion extends animatedObjects**

**{**

**public GreenfootImage[] explosion;**

**public ArrayList<Zombie> zombies;**

**/\*\***

**\* Act - do whatever the Explosion wants to do. This method is called whenever**

**\* the 'Act' or 'Run' button gets pressed in the environment.**

**\*/**

**public Explosion(ArrayList<Zombie> zombies) {**

**this.zombies = zombies;**

**explosion = importSprites("spudow",8);**

**}**

**public void addedToWorld(World world) {**

**Audio.play(80, "potato\_mine.mp3");**

**for (int i = zombies.size()-1; i >= 0; i--) {**

**if (Math.abs(zombies.get(i).getX() - getX()) < 44) {**

**zombies.get(i).takeDmg(900);**

**}**

**}**

**}**

**public void act()**

**{**

**if (frame <= 8) {**

**animate(explosion, 100L, false);**

**} else {**

**getWorld().removeObject(this);**

**return;**

**}**

**}**

**}**

###### 2.2.3.2.11.2.4 FallingObject

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class FallingObject extends animatedObjects**

**{**

**public double vSpeed = 0;**

**public double hSpeed = 0;**

**public double acceleration = 1;**

**public long deltaTime;**

**public long lastFrame = System.nanoTime();**

**public long currentFrame = System.nanoTime();**

**public long fallTime;**

**public int rotate;**

**public FallingObject(double vSpeed, double acceleration, double hSpeed, int rotate, long time) {**

**this.vSpeed = vSpeed;**

**this.acceleration = acceleration;**

**this.rotate = rotate;**

**this.hSpeed = hSpeed;**

**this.fallTime = time;**

**lastFrame = System.nanoTime();**

**currentFrame = System.nanoTime();**

**}**

**public void act()**

**{**

**update();**

**}**

**public void update() {**

**currentFrame = System.nanoTime();**

**deltaTime = (currentFrame - lastFrame) / 1000000;**

**if (deltaTime < fallTime) {**

**double x = getExactX()+hSpeed;**

**double y = getExactY()+vSpeed;**

**setLocation(x,y);**

**turn(rotate);**

**vSpeed = vSpeed + acceleration;**

**} else {**

**checkDeath();**

**}**

**}**

**public void checkDeath() {**

**if (getImage().getTransparency() > 0) {**

**if (getImage().getTransparency()-3 <= 0) {**

**getImage().setTransparency(0);**

**} else {**

**getImage().setTransparency(getImage().getTransparency()-3);**

**}**

**} else {**

**getWorld().removeObject(this);**

**return;**

**}**

**}**

**}**

###### 2.2.3.2.11.2.4.1 Arm

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Arm extends FallingObject**

**{**

**public Arm() {**

**super(0, 0.2, 0.2, RNG.Int(-1,1), 350L);**

**}**

**}**

###### 2.2.3.2.11.2.4.2 Brick

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Brick extends FallingObject{**

**public Brick() {**

**super(-2, 0.2, 0.9, RNG.Int(1, 5), 640L);**

**}**

**}**

###### 2.2.3.2.11.2.4.3 Bucket

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Bucket extends FallingObject**

**{**

**public Bucket() {**

**super(-2, 0.2, 0.9, RNG.Int(1, 5), 620L);**

**}**

**}**

###### 2.2.3.2.11.2.4.4 Cone

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Cone extends FallingObject{**

**public Cone() {**

**super(-2, 0.2, 0.9, RNG.Int(1, 5), 620L);**

**}**

**}**

###### 2.2.3.2.11.2.4.5 FallingSun

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class FallingSun extends FallingObject**

**{**

**public MyWorld MyWorld;**

**public GreenfootImage[] sun;**

**public boolean clicked = false;**

**public boolean beenClicked = false;**

**public long lastFrame2 = System.nanoTime();**

**public long deltaTime2 = System.nanoTime();**

**public FallingSun() {**

**super(0.6, 0, 0, 0, (long)RNG.Int(2000, 10000));**

**sun = importSprites("sun", 2);**

**}**

**public void update() {**

**currentFrame = System.nanoTime();**

**deltaTime = (currentFrame - lastFrame) / 1000000;**

**deltaTime2 = (currentFrame - lastFrame2) / 1000000;**

**animate(sun, 200, true);**

**if (!beenClicked) {**

**if (checkClick()) {**

**beenClicked = true;**

**Audio.play(90, "points.mp3");**

**MyWorld.seedbank.suncounter.addSun(25);**

**}**

**}**

**if (!beenClicked) {**

**if (deltaTime < fallTime) {**

**double x = getExactX()+hSpeed;**

**double y = getExactY()+vSpeed;**

**setLocation(x,y);**

**turn(rotate);**

**vSpeed = vSpeed + acceleration;**

**lastFrame2 = System.nanoTime();**

**} else {**

**if (deltaTime2 > 10000) {**

**if (getImage().getTransparency() > 0) {**

**if (getImage().getTransparency()-20 <= 0) {**

**getImage().setTransparency(0);**

**} else {**

**getImage().setTransparency(getImage().getTransparency()-20);**

**}**

**} else {**

**getWorld().removeObject(this);**

**return;**

**}**

**}**

**}**

**} else {**

**if (!(Math.abs(getX()-SunCounter.x) < 20 && Math.abs(getY()-SunCounter.y) < 20)) {**

**turnTowards(SunCounter.x,SunCounter.y);**

**move(20);**

**}**

**}**

**checkDeath();**

**}**

**public boolean checkClick() {**

**MouseInfo mouse = Greenfoot.getMouseInfo();**

**if (mouse != null && Greenfoot.mouseClicked(null)) {**

**MyWorld.moveHitbox();**

**if (intersects(MyWorld.hitbox)) {**

**clicked = true;**

**} else {**

**clicked = false;**

**}**

**return clicked;**

**}**

**clicked = false;**

**return clicked;**

**}**

**public void checkDeath() {**

**if (Math.abs(getX()-SunCounter.x) < 20 && Math.abs(getY()-SunCounter.y) < 20) {**

**move(0.5);**

**if (getImage().getTransparency() > 0) {**

**if (getImage().getTransparency()-20 <= 0) {**

**getImage().setTransparency(0);**

**} else {**

**getImage().setTransparency(getImage().getTransparency()-20);**

**}**

**} else {**

**getWorld().removeObject(this);**

**return;**

**}**

**}**

**}**

**@Override**

**public void addedToWorld(World world) {**

**MyWorld = (MyWorld)getWorld();**

**}**

**}**

###### 2.2.3.2.11.2.4.6 Head

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Head extends FallingObject{**

**public Head() {**

**super(-3, 0.2, 0.7, RNG.Int(1, 5), 700L);**

**}**

**}**

###### 2.2.3.2.11.2.4.7 Sun

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Sun extends FallingObject**

**{**

**public MyWorld MyWorld;**

**public GreenfootImage[] sun;**

**public boolean clicked = false;**

**public boolean beenClicked = false;**

**public long lastFrame2 = System.nanoTime();**

**public long deltaTime2 = System.nanoTime();**

**public Sun() {**

**super(-3, 0.15, RNG.Double(-100, 100), 0, 800L);**

**sun = importSprites("sun", 2);**

**}**

**public void update() {**

**currentFrame = System.nanoTime();**

**deltaTime = (currentFrame - lastFrame) / 1000000;**

**deltaTime2 = (currentFrame - lastFrame2) / 1000000;**

**animate(sun, 200, true);**

**if (!beenClicked) {**

**if (checkClick()) {**

**beenClicked = true;**

**Audio.play(90, "points.mp3");**

**MyWorld.seedbank.suncounter.addSun(25);**

**} else {**

**if (deltaTime2 > 12000) {**

**if (getImage().getTransparency() > 0) {**

**if (getImage().getTransparency()-20 <= 0) {**

**getImage().setTransparency(0);**

**} else {**

**getImage().setTransparency(getImage().getTransparency()-20);**

**}**

**} else {**

**getWorld().removeObject(this);**

**return;**

**}**

**}**

**}**

**}**

**if (!beenClicked) {**

**if (deltaTime < fallTime) {**

**double x = getExactX()+hSpeed;**

**double y = getExactY()+vSpeed;**

**setLocation(x,y);**

**turn(rotate);**

**vSpeed = vSpeed + acceleration;**

**}**

**} else {**

**if (!(Math.abs(getX()-SunCounter.x) < 20 && Math.abs(getY()-SunCounter.y) < 20)) {**

**turnTowards(SunCounter.x,SunCounter.y);**

**move(20);**

**}**

**}**

**checkDeath();**

**}**

**public boolean checkClick() {**

**MouseInfo mouse = Greenfoot.getMouseInfo();**

**if (mouse != null && Greenfoot.mouseClicked(null)) {**

**MyWorld.moveHitbox();**

**if (intersects(MyWorld.hitbox)) {**

**clicked = true;**

**} else {**

**clicked = false;**

**}**

**return clicked;**

**}**

**clicked = false;**

**return clicked;**

**}**

**public void checkDeath() {**

**if (Math.abs(getX()-SunCounter.x) < 20 && Math.abs(getY()-SunCounter.y) < 20) {**

**move(0.5);**

**if (getImage().getTransparency() > 0) {**

**if (getImage().getTransparency()-20 <= 0) {**

**getImage().setTransparency(0);**

**} else {**

**getImage().setTransparency(getImage().getTransparency()-20);**

**}**

**} else {**

**getWorld().removeObject(this);**

**return;**

**}**

**}**

**}**

**@Override**

**public void addedToWorld(World world) {**

**MyWorld = (MyWorld)getWorld();**

**}**

**}**

###### 2.2.3.2.11.2.4.8 WinCactus

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class WinCactus extends FallingObject**

**{**

**public GreenfootImage[] sprites = importSprites("cactuspacket", 2);**

**public boolean move = false;**

**public boolean addTrans = false;**

**public WinCactus() {**

**super(-3, 0.15, RNG.Double(-100, 100), 0, 750L);**

**}**

**public void update() {**

**currentFrame = System.nanoTime();**

**deltaTime = (currentFrame - lastFrame) / 1000000;**

**if (move) {**

**setImage("scactuspacket1.png");**

**if (getX() < (getWorld().getWidth()/2-5)) {**

**move(2);**

**} else if (getX() > (getWorld().getWidth()/2+5)){**

**move(-2);**

**} else {**

**if (!addTrans) {**

**addTrans=true;**

**getWorld().addObject(new Transition(false, new Level1(), "whitetransition.png", 1), 360, 215);**

**Audio.play(70, "lightfill.mp3");**

**}**

**}**

**} else {**

**animate(sprites, 200);**

**}**

**if (deltaTime < fallTime) {**

**double x = getExactX()+hSpeed;**

**double y = getExactY()+vSpeed;**

**setLocation(x,y);**

**turn(rotate);**

**vSpeed = vSpeed + acceleration;**

**} else {**

**}**

**if (Greenfoot.mouseClicked(this) && !move) {**

**move = true;**

**((MyWorld)getWorld()).finishLevel();**

**}**

**}**

**}**

###### 2.2.3.2.11.2.4.9 WinPotato

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class WinPotato extends FallingObject**

**{**

**public GreenfootImage[] sprites = importSprites("spotatopacket", 2);**

**public boolean move = false;**

**public boolean addTrans = false;**

**public WinPotato() {**

**super(-3, 0.15, RNG.Double(-100, 100), 0, 750L);**

**}**

**public void update() {**

**currentFrame = System.nanoTime();**

**deltaTime = (currentFrame - lastFrame) / 1000000;**

**if (move) {**

**setImage("spotatopacket1.png");**

**if (getX() < (getWorld().getWidth()/2-5)) {**

**move(2);**

**} else if (getX() > (getWorld().getWidth()/2+5)){**

**move(-2);**

**} else {**

**if (!addTrans) {**

**addTrans=true;**

**getWorld().addObject(new Transition(false, new Level1(), "whitetransition.png", 1), 360, 215);**

**Audio.play(70, "lightfill.mp3");**

**}**

**}**

**} else {**

**animate(sprites, 200);**

**}**

**if (deltaTime < fallTime) {**

**double x = getExactX()+hSpeed;**

**double y = getExactY()+vSpeed;**

**setLocation(x,y);**

**turn(rotate);**

**vSpeed = vSpeed + acceleration;**

**} else {**

**}**

**if (Greenfoot.mouseClicked(this) && !move) {**

**move = true;**

**((MyWorld)getWorld()).finishLevel();**

**}**

**}**

**}**

###### 2.2.3.2.11.2.4.10 WinRepeater

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class WinRepeater extends FallingObject**

**{**

**public GreenfootImage[] sprites = importSprites("srepeaterpacket", 2);**

**public boolean move = false;**

**public boolean addTrans = false;**

**public WinRepeater() {**

**super(-3, 0.15, RNG.Double(-100, 100), 0, 750L);**

**}**

**public void update() {**

**currentFrame = System.nanoTime();**

**deltaTime = (currentFrame - lastFrame) / 1000000;**

**if (move) {**

**setImage("srepeaterpacket1.png");**

**if (getX() < (getWorld().getWidth()/2-5)) {**

**move(2);**

**} else if (getX() > (getWorld().getWidth()/2+5)){**

**move(-2);**

**} else {**

**if (!addTrans) {**

**addTrans=true;**

**getWorld().addObject(new Transition(false, new Level2(), "whitetransition.png", 1), 360, 215);**

**Audio.play(70, "lightfill.mp3");**

**}**

**}**

**} else {**

**animate(sprites, 200);**

**}**

**if (deltaTime < fallTime) {**

**double x = getExactX()+hSpeed;**

**double y = getExactY()+vSpeed;**

**setLocation(x,y);**

**turn(rotate);**

**vSpeed = vSpeed + acceleration;**

**} else {**

**}**

**if (Greenfoot.mouseClicked(this) && !move) {**

**move = true;**

**((MyWorld)getWorld()).finishLevel();**

**}**

**}**

**}**

###### 2.2.3.2.11.2.5 IdleZombie

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class IdleZombie extends animatedObjects**

**{**

**/\*\***

**\* Act - do whatever the IdleZombie wants to do. This method is called whenever**

**\* the 'Act' or 'Run' button gets pressed in the environment.**

**\*/**

**public void act(){**

**}**

**public void addedToWorld(World world) {**

**frame = RNG.Int(1,4);**

**}**

**}**

###### 2.2.3.2.11.2.5.1 Basic

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Basic extends IdleZombie**

**{**

**/\*\***

**\* Act - do whatever the Basic wants to do. This method is called whenever**

**\* the 'Act' or 'Run' button gets pressed in the environment.**

**\*/**

**GreenfootImage[] idle;**

**public Basic() {**

**idle = importSprites("zombieidle", 4);**

**}**

**public void act()**

**{**

**animate(idle, 250, true);**

**// Add your action code here.**

**}**

**}**

###### 2.2.3.2.11.2.5.2 IdleBrickhead

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class IdleBrickhead extends IdleZombie**

**{**

**/\*\***

**\* Act - do whatever the Basic wants to do. This method is called whenever**

**\* the 'Act' or 'Run' button gets pressed in the environment.**

**\*/**

**GreenfootImage[] idle;**

**public IdleBrickhead() {**

**idle = importSprites("brickheadidle", 4);**

**}**

**public void act()**

**{**

**animate(idle, 300, true);**

**// Add your action code here.**

**}**

**}**

###### 2.2.3.2.11.2.5.3 IdleBucket

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class IdleBucket extends IdleZombie**

**{**

**/\*\***

**\* Act - do whatever the Basic wants to do. This method is called whenever**

**\* the 'Act' or 'Run' button gets pressed in the environment.**

**\*/**

**GreenfootImage[] idle;**

**public IdleBucket() {**

**idle = importSprites("bucketheadidle", 4);**

**}**

**public void act()**

**{**

**animate(idle, 300, true);**

**// Add your action code here.**

**}**

**}**

###### 2.2.3.2.11.2.5.4 IdleCone

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class IdleCone extends IdleZombie**

**{**

**/\*\***

**\* Act - do whatever the Basic wants to do. This method is called whenever**

**\* the 'Act' or 'Run' button gets pressed in the environment.**

**\*/**

**GreenfootImage[] idle;**

**public IdleCone() {**

**idle = importSprites("coneheadidle", 4);**

**}**

**public void act()**

**{**

**animate(idle, 200, true);**

**// Add your action code here.**

**}**

**}**

###### 2.2.3.2.11.2.6 Plant

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Plant extends animatedObjects {**

**public int maxHp;**

**public boolean isAlive = true;**

**public int hp;**

**public int damage;**

**public boolean opaque = false;**

**public MyWorld MyWorld;**

**public Plant() {**

**// Your initialization (if needed)**

**}**

**/\*\***

**\* Act - called each frame.**

**\* Before updating itself, the plant cleans up its row’s zombie list.**

**\*/**

**public void act() {**

**if (getWorld() != null) {**

**// Clean up any zombie references in this plant's row**

**cleanZombieRow();**

**if (isLiving()) {**

**update();**

**// Set transparency based on whether the plant is meant to be opaque**

**if (!opaque) {**

**getImage().setTransparency(255);**

**} else {**

**getImage().setTransparency(125);**

**}**

**} else {**

**MyWorld = (MyWorld)getWorld();**

**Audio.play(80, "gulp.mp3");**

**MyWorld.grid.removePlant(getXPos(), getYPos());**

**MyWorld.removeObject(this);**

**return;**

**}**

**}**

**}**

**/\*\***

**\* Clean up zombie references this plant's row in MyWorld.level.zombieRow**

**\* by removing any zombies that no longer exist in the world.**

**\* This helps ensure that when subclasses (e.g., Peashooter) iterate over their row,**

**\* they only process zombies that are actually present.**

**\*/**

**protected void cleanZombieRow() {**

**// Ensure we have a proper reference to MyWorld.**

**if(MyWorld == null) {**

**MyWorld = (MyWorld)getWorld();**

**if(MyWorld == null) {**

**return; // Still no world? Nothing to clean.**

**}**

**}**

**// Make sure that we have access to the zombieRow list.**

**if(MyWorld.level != null && MyWorld.level.zombieRow != null) {**

**// Retrieve the row that corresponds to this plant's grid position.**

**java.util.List<Zombie> row = MyWorld.level.zombieRow.get(getYPos());**

**// Iterate backward over the row list (so removals don't affect the loop)**

**for (int i = row.size() - 1; i >= 0; i--) {**

**Zombie z = row.get(i);**

**// If the zombie reference is null or no longer in the world, remove it.**

**if (z == null || z.getWorld() == null) {**

**row.remove(i);**

**}**

**}**

**}**

**}**

**// Default update() method to be overridden by subclasses.**

**public void update() {**

**// Plant-specific behavior lives in subclasses (e.g., Peashooter)**

**}**

**public int getXPos() {**

**return ((getX() - Grid.xOffset) / Grid.xSpacing);**

**}**

**public int getYPos() {**

**return ((getY() - Grid.yOffset) / Grid.ySpacing);**

**}**

**@Override**

**public void addedToWorld(World world) {**

**MyWorld = (MyWorld)getWorld();**

**MyWorld.addObject(new Dirt(), getX(), getY() + 30);**

**}**

**public boolean isLiving() {**

**if (hp <= 0) {**

**isAlive = false;**

**} else {**

**isAlive = true;**

**}**

**return isAlive;**

**}**

**public void hit(int dmg) {**

**// Default hit behavior (may be overridden)**

**}**

**}**

###### 2.2.3.2.11.2.6.1 Cactus

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Cactus extends Plant**

**{**

**private GreenfootImage[] idle;**

**private GreenfootImage[] shoot;**

**private boolean shootOnce = false;**

**private boolean shooting = false;**

**private long shootDelay = 2400L;**

**private long lastFrame2 = System.nanoTime();**

**private long deltaTime2;**

**public Cactus() {**

**maxHp = 100;**

**hp = maxHp;**

**shoot = importSprites("cactusshoot", 2);**

**idle = importSprites("cactus", 4);**

**}**

**public void hit(int dmg) {**

**if (isLiving()) {**

**if (!shootOnce) {**

**hitFlash(idle, "cactus");**

**} else {**

**hitFlash(shoot, "cactusshoot");**

**}**

**hp = hp-dmg;**

**}**

**}**

**public void update() {**

**MyWorld = (MyWorld)getWorld();**

**currentFrame = System.nanoTime();**

**if (!shooting) {**

**animate(idle, 150, true);**

**lastFrame2 = System.nanoTime();**

**} else {**

**deltaTime2 = (currentFrame - lastFrame2) / 1000000;**

**if (deltaTime2 < shootDelay) {**

**animate(idle, 150, true);**

**shootOnce = false;**

**} else {**

**if (!shootOnce) {**

**shootOnce = true;**

**frame = 0;**

**}**

**if (frame >= 2) {**

**MyWorld.addObject(new Needle(getYPos()), getX()+30,getY()-8);**

**lastFrame2 = currentFrame;**

**}**

**animate(shoot, 150, false);**

**}**

**}**

**if (MyWorld.level.zombieRow.get(getYPos()).size() == 0) {**

**shooting = false;**

**} else {**

**for (Zombie i : MyWorld.level.zombieRow.get(getYPos())) {**

**if (i.getX() > getX() && i.getX()<=MyWorld.getWidth()+10){**

**shooting = true;**

**break;**

**} else {**

**shooting = false;**

**}**

**}**

**}**

**}**

**}**

###### 2.2.3.2.11.2.6.2 Peashooter

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Peashooter extends Plant{**

**private GreenfootImage[] idle;**

**private GreenfootImage[] shoot;**

**private boolean shootOnce = false;**

**private boolean shooting = false;**

**private long shootDelay = 1700L;**

**private long lastFrame2 = System.nanoTime();**

**private long deltaTime2;**

**public Peashooter() {**

**maxHp = 60;**

**hp = maxHp;**

**shoot = importSprites("peashootershoot", 3);**

**idle = importSprites("peashooter", 9);**

**}**

**public void hit(int dmg) {**

**if (isLiving()) {**

**if (!shootOnce) {**

**hitFlash(idle, "peashooter");**

**} else {**

**hitFlash(shoot, "peashootershoot");**

**}**

**hp = hp-dmg;**

**}**

**}**

**public void update() {**

**MyWorld = (MyWorld)getWorld();**

**currentFrame = System.nanoTime();**

**if (!shooting) {**

**animate(idle, 150, true);**

**lastFrame2 = System.nanoTime();**

**} else {**

**deltaTime2 = (currentFrame - lastFrame2) / 1000000;**

**if (deltaTime2 < shootDelay) {**

**animate(idle, 150, true);**

**shootOnce = false;**

**} else {**

**if (!shootOnce) {**

**shootOnce = true;**

**frame = 0;**

**}**

**if (frame >= 3) {**

**Audio.play(80, "throw.mp3", "throw2.mp3");**

**MyWorld.addObject(new Pea(getYPos()), getX()+25,getY()-17);**

**lastFrame2 = currentFrame;**

**}**

**animate(shoot, 100, false);**

**}**

**}**

**if (MyWorld.level.zombieRow.get(getYPos()).size() == 0) {**

**shooting = false;**

**} else {**

**for (Zombie i : MyWorld.level.zombieRow.get(getYPos())) {**

**if (i.getX() > getX() && i.getX()<=MyWorld.getWidth()+10){**

**shooting = true;**

**break;**

**} else {**

**shooting = false;**

**}**

**}**

**}**

**}**

**}**

###### 2.2.3.2.11.2.6.3 Potatomine

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class PotatoMine extends Plant**

**{**

**private GreenfootImage[] idle;**

**private GreenfootImage[] arm;**

**private boolean playOnce = false;**

**public boolean armed = false;**

**private long lastFrame2 = System.nanoTime();**

**private boolean playSFX = false;**

**private long deltaTime2 = System.nanoTime();**

**public PotatoMine() {**

**idle = importSprites("potato", 5);**

**arm = importSprites("potatomine", 3);**

**maxHp = 60;**

**hp = maxHp;**

**currentFrame = System.nanoTime();**

**lastFrame2 = System.nanoTime();**

**}**

**public void update() {**

**currentFrame = System.nanoTime();**

**deltaTime2 = (currentFrame - lastFrame2) / 1000000;**

**if (deltaTime2 > 22000L) {**

**armed = true;**

**if (!playOnce) {**

**if (!playSFX) {**

**playSFX = true;**

**Audio.play(70, "dirt\_rise.mp3");**

**}**

**animate(arm, 200, false);**

**if (frame > 2) {**

**playOnce = true;**

**}**

**} else {**

**animate(idle, 200, true);**

**}**

**checkExplosion();**

**}**

**}**

**public void hit(int dmg) {**

**if (isLiving() && !armed) {**

**hp = hp-dmg;**

**}**

**}**

**public void checkExplosion() {**

**if (MyWorld.level.zombieRow.get(getYPos()).size() == 0) {**

**} else {**

**for (Zombie i : MyWorld.level.zombieRow.get(getYPos())) {**

**if (Math.abs(i.getX() - getX()) < 28) {**

**getWorld().addObject(new Explosion(MyWorld.level.zombieRow.get(getYPos())), getX(), getY()-25);**

**((MyWorld)getWorld()).grid.removePlant(getXPos(), getYPos());**

**return;**

**}**

**}**

**}**

**}**

**}**

###### 2.2.3.2.11.2.6.4 Repeater

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Repeater extends Plant {**

**private GreenfootImage[] idle;**

**private GreenfootImage[] shoot;**

**private boolean shootOnce = false;**

**private int shootCount = 0;**

**private boolean resetFrame = false;**

**private boolean shooting = false;**

**private long shootDelay = 1700L;**

**private long lastFrame2 = System.nanoTime();**

**private long deltaTime2;**

**public Repeater() {**

**maxHp = 60;**

**hp = maxHp;**

**shoot = importSprites("repeatershoot", 3);**

**idle = importSprites("repeater", 7);**

**}**

**public void hit(int dmg) {**

**if (isLiving()) {**

**if (!shootOnce) {**

**hitFlash(idle, "repeater");**

**} else {**

**hitFlash(shoot, "repeatershoot");**

**}**

**hp = hp-dmg;**

**}**

**}**

**public void update() {**

**MyWorld = (MyWorld)getWorld();**

**currentFrame = System.nanoTime();**

**if (!shooting) {**

**animate(idle, 225, true);**

**lastFrame2 = System.nanoTime();**

**} else {**

**deltaTime2 = (currentFrame - lastFrame2) / 1000000;**

**if (deltaTime2 < shootDelay) {**

**animate(idle, 225, true);**

**shootCount = 0;**

**resetFrame = false;**

**} else {**

**if (shootCount >= 2) {**

**lastFrame2 = currentFrame;**

**}**

**if (!resetFrame) {**

**setFrame(1);**

**resetFrame = true;**

**}**

**if (frame >= 3) {**

**Audio.play(80, "throw.mp3", "throw2.mp3");**

**MyWorld.addObject(new Pea(getYPos()), getX()+25,getY()-17);**

**setFrame(1);**

**setImage("repeatershoot1.png");**

**shootCount++;**

**}**

**animate(shoot, 70, false);**

**}**

**}**

**if (MyWorld.level.zombieRow.get(getYPos()).size() == 0) {**

**shooting = false;**

**} else {**

**for (Zombie i : MyWorld.level.zombieRow.get(getYPos())) {**

**if (i.getX() > getX() && i.getX()<=MyWorld.getWidth()+10){**

**shooting = true;**

**break;**

**} else {**

**shooting = false;**

**}**

**}**

**}**

**}**

**}**

###### 2.2.3.2.11.2.6.5 Sunflower

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Sunflower extends Plant{**

**private GreenfootImage[] idle;**

**private long lastFrame2 = System.nanoTime();**

**private long deltaTime2;**

**private boolean test = false;**

**public Sunflower(){**

**idle = importSprites("sunfloweridle", 8);**

**maxHp = 60;**

**hp = maxHp;**

**}**

**public void update(){**

**produceSun();**

**animate(idle, 200, true);**

**}**

**public void hit(int dmg) {**

**if (isLiving()) {**

**hitFlash(idle, "sunfloweridle");**

**hp = hp-dmg;**

**}**

**}**

**public void produceSun() {**

**deltaTime2 = (currentFrame - lastFrame2) / 1000000;**

**if (deltaTime2 > 20000L) {**

**lastFrame2 = System.nanoTime();**

**hitFlash(idle, "sunfloweridle");**

**test= true;**

**MyWorld.addObject(new Sun(), getX(), getY()-10);**

**}**

**}**

**}**

###### 2.2.3.2.11.2.6.6 Walnut

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**/\*\***

**\* Write a description of class Walnut here.**

**\***

**\* @author (your name)**

**\* @version (a version number or a date)**

**\*/**

**public class Walnut extends Plant**

**{**

**private GreenfootImage[] idle;**

**private GreenfootImage[] d;**

**private GreenfootImage[] dd;**

**private boolean test = false;**

**private long lastFrame2 = System.nanoTime();**

**private long deltaTime2;**

**public Walnut() {**

**idle = importSprites("walnut", 5);**

**d = importSprites("walnutd", 5);**

**dd = importSprites("walnutdd", 5);**

**maxHp = 730;**

**hp = maxHp;**

**}**

**public void update(){**

**if (hp >= 480) {**

**animate(idle, 200, true);**

**} else if (hp >= 240) {**

**animate(d, 200, true);**

**} else {**

**animate(dd, 200, true);**

**}**

**}**

**public void hit(int dmg) {**

**if (isLiving()) {**

**if (hp >= 480) {**

**hitFlash(idle, "walnut");**

**} else if (hp >= 240) {**

**hitFlash(d, "walnutd");**

**} else {**

**hitFlash(dd, "walnutdd");**

**}**

**hp = hp-dmg;**

**}**

**}**

**}**

###### 2.2.3.2.11.2.7 Projectile

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**/\*\***

**\* Write a description of class Pea here.**

**\***

**\* @author (your name)**

**\* @version (a version number or a date)**

**\*/**

**public class Projectile extends animatedObjects{**

**public int speed = 4;**

**public GreenfootImage[] image;**

**public boolean hit = false;**

**public MyWorld MyWorld;**

**public boolean foundTarget = false;**

**public Zombie hitZombie;**

**public int frameCount;**

**public int yPos;**

**public int damage;**

**/\*\***

**\* Act - do whatever the Pea wants to do. This method is called whenever**

**\* the 'Act' or 'Run' button gets pressed in the environment.**

**\*/**

**public Projectile(String name, int frameCount, int yPos, int dmg, int speed) {**

**this.frameCount = frameCount;**

**this.image = importSprites(name, frameCount);**

**this.yPos = yPos;**

**this.damage = dmg;**

**this.speed = speed;**

**setImage(image[0]);**

**}**

**public void act()**

**{**

**if (getWorld() != null) {**

**MyWorld = (MyWorld)getWorld();**

**if (frame >= frameCount) {**

**MyWorld.removeObject(this);**

**return;**

**}**

**if (!hit) {**

**move(speed);**

**} else {**

**animate(image, 150, false);**

**}**

**if (isAtEdge()) {**

**MyWorld.removeObject(this);**

**return;**

**}**

**for (Zombie i : MyWorld.level.zombieRow.get(yPos)) {**

**if (Math.abs(i.getX() - getX()) < 30) {**

**if (!foundTarget) {**

**hitZombie = i;**

**foundTarget = true;**

**}**

**if (!hit) {**

**hitZombie.hit(damage);**

**hit = true;**

**} else if (hitZombie.getWorld() != null && getX() < hitZombie.getX()) {**

**move(speed);**

**}**

**}**

**}**

**}**

**}**

**}**

###### 2.2.3.2.11.2.7.1 Needle

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Needle extends Projectile**

**{**

**public Needle(int yPos) {**

**super("needle", 1, yPos, 15, 6);**

**}**

**}**

###### 2.2.3.2.11.2.7.2 Pea

**import greenfoot.\*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)**

**public class Pea extends Projectile{**

**/\*\***

**\* Act - do whatever the Pea wants to do. This method is called whenever**

**\* the 'Act' or 'Run' button gets pressed in the environment.**

**\*/**

**public Pea(int yPos) {**

**super("pea", 3, yPos, 10, 4);**

**}**

**}**

###### 2.2.3.2.11.2.8 ReadySetPlant

**import** greenfoot**.\*;** // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

public class ReadySetPlant **extends** animatedObjects

**{**

/\*\*

\* Act - do whatever the ReadySetPlant wants to do. This method is called whenever

\* the 'Act' or 'Run' button gets pressed in the environment.

\*/

public GreenfootImage**[]** ready**;**

public GreenfootImage**[]** set**;**

public GreenfootImage plant**;**

public boolean r **=** **true;**

public boolean s **=** **true;**

public boolean p **=** **true;**

public int counter**;**

public ReadySetPlant**()** **{**

ready **=** importSprites**(**"Ready\_\_"**,**11**);**

set **=** importSprites**(**"Set\_\_"**,**12**);**

plant **=** **new** GreenfootImage**(**"PLANT!.png"**);**

**}**

public void act**()**

**{**

**if** **(**frame **<=** 11 **&&** counter **<** 60**)** **{**

**if** **(**r**)** **{**

animate**(**ready**,** 40**,** **false);**

**}** **else** **if** **(**s**)** **{**

animate**(**set**,** 40**,** **false);**

**}** **else** **if** **(**p**)** **{**

setImage**(**plant**);**

counter**++;**

**}**

**}** **else** **{**

**if** **(**r**)** **{**

r **=** **false;**

frame **=** 0**;**

**}** **else** **if** **(**s**)** **{**

s **=** **false;**

frame **=** 0**;**

**}** **else** **if** **(**p**)** **{**

p **=** **false;**

getWorld**().**removeObject**(this);**

**return;**

**}**

**}**

**}**

**}**

###### 2.2.3.2.11.2.9 Zombie

**import** greenfoot**.\*;** // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

**import** java**.**util**.\*;**

public class Zombie **extends** animatedObjects

**{**

public boolean fallen **=** **false;**

public boolean eating **=** **false;**

public boolean eatOnce **=** **false;**

public int hp**;**

public int maxHp**;**

public double walkSpeed**;**

public MyWorld MyWorld**;**

public boolean spawnHead **=** **false;**

public Plant target**;**

public int eatSpeed**;**

public boolean isAlive **=** **true;**

public GreenfootImage**[]** headless**;**

public GreenfootImage**[]** headlesseating**;**

public GreenfootImage**[]** fall**;**

public boolean resetAnim **=** **false;**

public boolean finalDeath **=** **false;**

public boolean fixAnim **=** **false;**

/\*\*

\* Act - do whatever the Zombie wants to do. This method is called whenever

\* the 'Act' or 'Run' button gets pressed in the environment.

\*/

public Zombie**()** **{**

headless **=** importSprites**(**"zombieheadless"**,** 7**);**

fall **=** importSprites**(**"zombiefall"**,**6**);**

headlesseating **=** importSprites**(**"headlesszombieeating"**,** 7**);**

**}**

public void act**()**

**{**

**if** **(**getWorld**()** **!=** **null)** **{**

**if** **(**isLiving**())** **{**

update**();**

**}** **else** **{**

deathAnim**();**

**}**

**}**

**}**

public void update**()** **{**

**}**

public void deathAnim**()** **{**

**if** **(!**resetAnim**)** **{**

frame **=** 0**;**

resetAnim **=** **true;**

**}**

**if** **(**frame **<=**6**)** **{**

**if** **(**finalDeath**)** **{**

**if** **(!**fixAnim**)** **{**

fixAnim **=** **true;**

Audio**.**play**(**80**,** "zombie\_falling\_1.mp3"**,** "zombie\_falling\_2.mp3"**);**

MyWorld**.**addObject**(new** fallingZombie**(**fall**),** getX**()-**12**,** getY**()+**20**);**

MyWorld**.**removeObject**(this);**

**return;**

**}**

**}** **else** **{**

**if** **(!**spawnHead**)** **{**

spawnHead **=** **true;**

Audio**.**play**(**80**,** "limbs\_pop.mp3"**);**

getWorld**().**addObject**(new** Head**(),** getX**(),** getY**()-**10**);**

**}**

**if** **(!**eating**)** **{**

animate**(**headless**,** 350**,** **false);**

move**(-**walkSpeed**);**

**}** **else** **{**

animate**(**headlesseating**,** 350**,** **false);**

**}**

**}**

**}** **else** **if** **(!**finalDeath**)** **{**

resetAnim **=** **false;**

finalDeath **=** **true;**

**for** **(**ArrayList**<**Zombie**>** i **:** MyWorld**.**level**.**zombieRow**)** **{**

**if** **(**i**.**contains**(this))** **{**

i**.**remove**(this);**

**break;**

**}**

**}**

**}**

**}**

public void playEating**()** **{**

**if** **(**frame **==** 5 **||** frame **==** 2**)** **{**

**if** **(!**eatOnce**)** **{**

eatOnce **=** **true;**

Audio**.**play**(**70**,** "chomp.mp3"**,** "chomp2.mp3"**,** "chompsoft.mp3"**);**

target**.**hit**(**10**);**

**}**

**}** **else** **{**

eatOnce **=** **false;**

**}**

**}**

@Override

protected void addedToWorld**(**World world**)** **{**

MyWorld **=** **(**MyWorld**)**getWorld**();**

**}**

public boolean isLiving**()** **{**

**if** **(**hp **<=**0**)** **{**

isAlive **=** **false;**

**}** **else** **{**

isAlive **=** **true;**

**}**

**return** isAlive**;**

**}**

public void hit**(**int dmg**)** **{**

**}**

public void takeDmg**(**int dmg**)** **{**

hp **-=** dmg**;**

**if** **(**hp **<=** 0**)** **{**

**for** **(**ArrayList**<**Zombie**>** i **:** MyWorld**.**level**.**zombieRow**)** **{**

**if** **(**i**.**contains**(this))** **{**

i**.**remove**(this);**

**break;**

**}**

**}**

getWorld**().**removeObject**(this);**

**return;**

**}**

**}**

public boolean isEating**()** **{**

var row **=** MyWorld**.**grid**.**Grid**[**getYPos**()];**

**for** **(**int i **=** 0**;** i **<** MyWorld**.**grid**.**Grid**[**0**].**length**;** i**++)** **{**

**if** **(**row**[**i**]** **!=** **null)** **{**

**if** **(**Math**.**abs**(**row**[**i**].**getX**()** **-** getX**()+**5**)** **<** 35**)** **{**

**if** **(**row**[**i**]** **instanceof** PotatoMine**)** **{**

**if** **(((**PotatoMine**)**row**[**i**]).**armed **==** **true)** **{**

eating **=** **false;**

**return** **false;**

**}**

**}**

eating **=** **true;**

target **=** row**[**i**];**

**return** eating**;**

**}**

**}**

**}**

eating **=** **false;**

**return** eating**;**

**}**

public int getYPos**()** **{**

**return** **((**getY**()-**MyWorld**.**level**.**yOffset**)/**MyWorld**.**level**.**ySpacing**);**

**}**

public int getXPos**()** **{**

**return** getX**();**

**}**

**}**

###### 2.2.3.2.11.2.9.1 BasicZombie

**import** greenfoot**.\*;** // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

public class BasicZombie **extends** Zombie

**{**

public GreenfootImage**[]** idle**;**

public GreenfootImage**[]** walk**;**

public GreenfootImage**[]** armless**;**

public GreenfootImage**[]** eat**;**

public GreenfootImage**[]** armlesseat**;**

public BasicZombie**()** **{**

walk **=** importSprites**(**"zombiewalk"**,** 7**);**

eat **=** importSprites**(**"zombieeating"**,** 7**);**

armlesseat **=** importSprites**(**"armlesszombieeating"**,** 7**);**

armless **=** importSprites**(**"armlesszombie"**,** 7**);**

walkSpeed **=** RNG**.**Double**(**11**,** 14**);**

maxHp **=** 100**;**

hp **=** maxHp**;**

**}**

public void update**()** **{**

**if** **(**hp **>** 50**)** **{**

**if** **(!**isEating**())** **{**

animate**(**walk**,** 350**,** **true);**

move**(-**walkSpeed**);**

**}** **else** **{**

animate**(**eat**,** 200**,** **true);**

playEating**();**

**}**

**}** **else** **{**

**if** **(!**fallen**)** **{**

fallen **=** **true;**

Audio**.**play**(**80**,** "limbs\_pop.mp3"**);**

MyWorld**.**addObject**(new** Arm**(),** getX**()+**8**,** getY**()+**20**);**

**}**

**if** **(!**isEating**())** **{**

animate**(**armless**,** 350**,** **true);**

move**(-**walkSpeed**);**

**}** **else** **{**

animate**(**armlesseat**,** 200**,** **true);**

playEating**();**

**}**

**}**

**}**

public void hit**(**int dmg**)** **{**

Audio**.**play**(**70**,** "splat.mp3"**,** "splat2.mp3"**,** "splat3.mp3"**);**

**if** **(**isLiving**())** **{**

**if** **(!**fallen**)** **{**

**if** **(!**eating**)** **{**

hitFlash**(**walk**,** "zombiewalk"**);**

**}** **else** **{**

hitFlash**(**eat**,** "zombieeating"**);**

**}**

**}** **else** **{**

**if** **(!**eating**)** **{**

hitFlash**(**armless**,** "armlesszombie"**);**

**}** **else** **{**

hitFlash**(**armlesseat**,** "armlesszombieeating"**);**

**}**

**}**

hp **-=** dmg**;**

**}** **else** **if** **(!**finalDeath**)** **{**

**if** **(!**eating**)** **{**

hitFlash**(**headless**,** "zombieheadless"**);**

**}** **else** **{**

hitFlash**(**headlesseating**,** "headlesszombieeating"**);**

**}**

**}**

**}**

**}**

###### 2.2.3.2.11.2.9.2 Brickhead

**import** greenfoot**.\*;** // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

public class Brickhead **extends** Zombie

**{**

public boolean brick **=** **true;**

public GreenfootImage**[]** idle**;**

public GreenfootImage**[]** walk**;**

public GreenfootImage**[]** armless**;**

public GreenfootImage**[]** eat**;**

public GreenfootImage**[]** armlesseat**;**

public GreenfootImage**[]** brickhead**;**

public GreenfootImage**[]** brickheadwalk**;**

public GreenfootImage**[]** brickheadwalkd**;**

public GreenfootImage**[]** brickheadwalkdd**;**

public GreenfootImage**[]** brickheadeat**;**

public GreenfootImage**[]** brickheadeatd**;**

public GreenfootImage**[]** brickheadeatdd**;**

public Brickhead**()** **{**

idle **=** importSprites**(**"zombieidle"**,** 4**);**

walk **=** importSprites**(**"zombiewalk"**,** 7**);**

eat **=** importSprites**(**"zombieeating"**,** 7**);**

armlesseat **=** importSprites**(**"armlesszombieeating"**,** 7**);**

armless **=** importSprites**(**"armlesszombie"**,** 7**);**

brickheadwalk **=** importSprites**(**"brickhead"**,** 7**);**

brickheadwalkd **=** importSprites**(**"brickheadd"**,** 7**);**

brickheadwalkdd **=** importSprites**(**"brickheaddd"**,** 7**);**

brickheadeat **=** importSprites**(**"brickheadeat"**,** 7**);**

brickheadeatd **=** importSprites**(**"brickheadeatd"**,** 7**);**

brickheadeatdd **=** importSprites**(**"brickheadeatdd"**,** 7**);**

walkSpeed **=** RNG**.**Double**(**11**,** 14**);**

maxHp **=** 1400**;**

hp **=** maxHp**;**

**}**

public void update**()** **{**

**if** **(**hp **>** 966**)** **{**

**if** **(!**isEating**())** **{**

animate**(**brickheadwalk**,** 350**,** **true);**

move**(-**walkSpeed**);**

**}** **else** **{**

animate**(**brickheadeat**,** 200**,** **true);**

playEating**();**

**}**

**}** **else** **if** **(**hp **>** 533**)** **{**

**if** **(!**isEating**())** **{**

animate**(**brickheadwalkd**,** 350**,** **true);**

move**(-**walkSpeed**);**

**}** **else** **{**

animate**(**brickheadeatd**,** 200**,** **true);**

playEating**();**

**}**

**}** **else** **if** **(**hp **>** 100**)** **{**

**if** **(!**isEating**())** **{**

animate**(**brickheadwalkdd**,** 350**,** **true);**

move**(-**walkSpeed**);**

**}** **else** **{**

animate**(**brickheadeatdd**,** 200**,** **true);**

playEating**();**

**}**

**}** **else** **{**

**if** **(**brick**)** **{**

brick **=** **false;**

MyWorld**.**addObject**(new** Brick**(),** getX**(),** getY**()-**25**);**

**}**

**if** **(**hp **>** 50**)** **{**

**if** **(!**isEating**())** **{**

animate**(**walk**,** 350**,** **true);**

move**(-**walkSpeed**);**

**}** **else** **{**

animate**(**eat**,** 200**,** **true);**

playEating**();**

**}**

**}** **else** **{**

**if** **(!**fallen**)** **{**

fallen **=** **true;**

Audio**.**play**(**80**,** "limbs\_pop.mp3"**);**

MyWorld**.**addObject**(new** Arm**(),** getX**()+**8**,** getY**()+**20**);**

**}**

**if** **(!**isEating**())** **{**

animate**(**armless**,** 350**,** **true);**

move**(-**walkSpeed**);**

**}** **else** **{**

animate**(**armlesseat**,** 200**,** **true);**

playEating**();**

**}**

**}**

**}**

**}**

public void hit**(**int dmg**)** **{**

Audio**.**play**(**70**,** "splat.mp3"**,** "splat2.mp3"**,** "splat3.mp3"**);**

**if** **(**isLiving**())** **{**

**if** **(**hp **>** 966**)** **{**

**if** **(!**isEating**())** **{**

hitFlash**(**brickheadwalk**,** "brickhead"**);**

**}** **else** **{**

hitFlash**(**brickheadeat**,** "brickheadeat"**);**

**}**

**}** **else** **if** **(**hp **>** 533**)** **{**

**if** **(!**isEating**())** **{**

hitFlash**(**brickheadwalkd**,** "brickheadd"**);**

**}** **else** **{**

hitFlash**(**brickheadeatd**,** "brickheadeatd"**);**

**}**

**}** **else** **if** **(**hp **>** 100**)** **{**

**if** **(!**isEating**())** **{**

hitFlash**(**brickheadwalkdd**,** "brickheaddd"**);**

**}** **else** **{**

hitFlash**(**brickheadeatdd**,** "brickheadeatdd"**);**

**}**

**}** **else** **{**

**if** **(!**fallen**)** **{**

**if** **(!**eating**)** **{**

hitFlash**(**walk**,** "zombiewalk"**);**

**}** **else** **{**

hitFlash**(**eat**,** "zombieeating"**);**

**}**

**}** **else** **{**

**if** **(!**eating**)** **{**

hitFlash**(**armless**,** "armlesszombie"**);**

**}** **else** **{**

hitFlash**(**armlesseat**,** "armlesszombieeating"**);**

**}**

**}**

**}**

hp **-=** dmg**;**

**}** **else** **if** **(!**finalDeath**)** **{**

**if** **(!**eating**)** **{**

hitFlash**(**headless**,** "zombieheadless"**);**

**}** **else** **{**

hitFlash**(**headlesseating**,** "headlesszombieeating"**);**

**}**

**}**

**}**

**}**

###### 2.2.3.2.11.2.9.3 Buckethead

**import** greenfoot**.\*;** // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

/\*\*

\* Write a description of class BasicZombie here.

\*

\* **@author** (your name)

\* **@version** (a version number or a date)

\*/

public class Buckethead **extends** Zombie

**{**

/\*\*

\* Act - do whatever the BasicZombie wants to do. This method is called whenever

\* the 'Act' or 'Run' button gets pressed in the environment.

\*/

public boolean bucket **=** **true;**

public GreenfootImage**[]** idle**;**

public GreenfootImage**[]** walk**;**

public GreenfootImage**[]** armless**;**

public GreenfootImage**[]** eat**;**

public GreenfootImage**[]** armlesseat**;**

public GreenfootImage**[]** buckethead**;**

public GreenfootImage**[]** bucketheadwalk**;**

public GreenfootImage**[]** bucketheadwalkd**;**

public GreenfootImage**[]** bucketheadwalkdd**;**

public GreenfootImage**[]** bucketheadeat**;**

public GreenfootImage**[]** bucketheadeatd**;**

public GreenfootImage**[]** bucketheadeatdd**;**

public Buckethead**()** **{**

idle **=** importSprites**(**"zombieidle"**,** 4**);**

walk **=** importSprites**(**"zombiewalk"**,** 7**);**

eat **=** importSprites**(**"zombieeating"**,** 7**);**

armlesseat **=** importSprites**(**"armlesszombieeating"**,** 7**);**

armless **=** importSprites**(**"armlesszombie"**,** 7**);**

//Conehead

bucketheadwalk **=** importSprites**(**"bucketheadwalk"**,** 7**);**

bucketheadwalkd **=** importSprites**(**"bucketheadwalkd"**,** 7**);**

bucketheadwalkdd **=** importSprites**(**"bucketheadwalkdd"**,** 7**);**

bucketheadeat **=** importSprites**(**"bucketheadeat"**,** 7**);**

bucketheadeatd **=** importSprites**(**"bucketheadeatd"**,** 7**);**

bucketheadeatdd **=** importSprites**(**"bucketheadeatdd"**,** 7**);**

walkSpeed **=** RNG**.**Double**(**11**,** 14**);**

maxHp **=** 700**;**

hp **=** maxHp**;**

**}**

public void update**()** **{**

**if** **(**hp **>** 500**)** **{**

**if** **(!**isEating**())** **{**

animate**(**bucketheadwalk**,** 350**,** **true);**

move**(-**walkSpeed**);**

**}** **else** **{**

animate**(**bucketheadeat**,** 200**,** **true);**

playEating**();**

**}**

**}** **else** **if** **(**hp **>** 300**)** **{**

**if** **(!**isEating**())** **{**

animate**(**bucketheadwalkd**,** 350**,** **true);**

move**(-**walkSpeed**);**

**}** **else** **{**

animate**(**bucketheadeatd**,** 200**,** **true);**

playEating**();**

**}**

**}** **else** **if** **(**hp **>** 100**)** **{**

**if** **(!**isEating**())** **{**

animate**(**bucketheadwalkdd**,** 350**,** **true);**

move**(-**walkSpeed**);**

**}** **else** **{**

animate**(**bucketheadeatdd**,** 200**,** **true);**

playEating**();**

**}**

**}** **else** **{**

**if** **(**bucket**)** **{**

bucket **=** **false;**

MyWorld**.**addObject**(new** Bucket**(),** getX**(),** getY**()-**25**);**

**}**

**if** **(**hp **>** 50**)** **{**

**if** **(!**isEating**())** **{**

animate**(**walk**,** 350**,** **true);**

move**(-**walkSpeed**);**

**}** **else** **{**

animate**(**eat**,** 200**,** **true);**

playEating**();**

**}**

**}** **else** **{**

**if** **(!**fallen**)** **{**

fallen **=** **true;**

Audio**.**play**(**80**,** "limbs\_pop.mp3"**);**

MyWorld**.**addObject**(new** Arm**(),** getX**()+**8**,** getY**()+**20**);**

**}**

**if** **(!**isEating**())** **{**

animate**(**armless**,** 350**,** **true);**

move**(-**walkSpeed**);**

**}** **else** **{**

animate**(**armlesseat**,** 200**,** **true);**

playEating**();**

**}**

**}**

**}**

**}**

public void hit**(**int dmg**)** **{**

**if** **(**bucket**)** **{**

Audio**.**play**(**70**,** "shieldhit.mp3"**,** "shieldhit2.mp3"**);**

**}** **else** **{**

Audio**.**play**(**70**,** "splat.mp3"**,** "splat2.mp3"**,** "splat3.mp3"**);**

**}**

**if** **(**isLiving**())** **{**

**if** **(**hp **>** 500**)** **{**

**if** **(!**isEating**())** **{**

hitFlash**(**bucketheadwalk**,** "bucketheadwalk"**);**

**}** **else** **{**

hitFlash**(**bucketheadeat**,** "bucketheadeat"**);**

**}**

**}** **else** **if** **(**hp **>** 300**)** **{**

**if** **(!**isEating**())** **{**

hitFlash**(**bucketheadwalkd**,** "bucketheadwalkd"**);**

**}** **else** **{**

hitFlash**(**bucketheadeatd**,** "bucketheadeatd"**);**

**}**

**}** **else** **if** **(**hp **>** 100**)** **{**

**if** **(!**isEating**())** **{**

hitFlash**(**bucketheadwalkdd**,** "bucketheadwalkdd"**);**

**}** **else** **{**

hitFlash**(**bucketheadeatdd**,** "bucketheadeatdd"**);**

**}**

**}** **else** **{**

**if** **(!**fallen**)** **{**

**if** **(!**eating**)** **{**

hitFlash**(**walk**,** "zombiewalk"**);**

**}** **else** **{**

hitFlash**(**eat**,** "zombieeating"**);**

**}**

**}** **else** **{**

**if** **(!**eating**)** **{**

hitFlash**(**armless**,** "armlesszombie"**);**

**}** **else** **{**

hitFlash**(**armlesseat**,** "armlesszombieeating"**);**

**}**

**}**

**}**

hp **-=** dmg**;**

**}** **else** **if** **(!**finalDeath**)** **{**

**if** **(!**eating**)** **{**

hitFlash**(**headless**,** "zombieheadless"**);**

**}** **else** **{**

hitFlash**(**headlesseating**,** "headlesszombieeating"**);**

**}**

**}**

**}**

**}**

###### 2.2.3.2.11.2.9.4 Conehead

**import** greenfoot**.\*;** // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

public class Conehead **extends** Zombie

**{**

public boolean cone **=** **true;**

public GreenfootImage**[]** idle**;**

public GreenfootImage**[]** walk**;**

public GreenfootImage**[]** armless**;**

public GreenfootImage**[]** eat**;**

public GreenfootImage**[]** armlesseat**;**

public GreenfootImage**[]** conehead**;**

public GreenfootImage**[]** coneheadwalk**;**

public GreenfootImage**[]** coneheadwalkd**;**

public GreenfootImage**[]** coneheadwalkdd**;**

public GreenfootImage**[]** coneheadeat**;**

public GreenfootImage**[]** coneheadeatd**;**

public GreenfootImage**[]** coneheadeatdd**;**

public Conehead**()** **{**

idle **=** importSprites**(**"zombieidle"**,** 4**);**

walk **=** importSprites**(**"zombiewalk"**,** 7**);**

eat **=** importSprites**(**"zombieeating"**,** 7**);**

armlesseat **=** importSprites**(**"armlesszombieeating"**,** 7**);**

armless **=** importSprites**(**"armlesszombie"**,** 7**);**

//Conehead

coneheadwalk **=** importSprites**(**"coneheadwalk"**,** 7**);**

coneheadwalkd **=** importSprites**(**"coneheadwalkd"**,** 7**);**

coneheadwalkdd **=** importSprites**(**"coneheadwalkdd"**,** 7**);**

coneheadeat **=** importSprites**(**"coneheadeat"**,** 7**);**

coneheadeatd **=** importSprites**(**"coneheadeatd"**,** 7**);**

coneheadeatdd **=** importSprites**(**"coneheadeatdd"**,** 7**);**

walkSpeed **=** RNG**.**Double**(**11**,** 14**);**

maxHp **=** 300**;**

hp **=** maxHp**;**

**}**

public void update**()** **{**

**if** **(**hp **>** 232**)** **{**

**if** **(!**isEating**())** **{**

animate**(**coneheadwalk**,** 350**,** **true);**

move**(-**walkSpeed**);**

**}** **else** **{**

animate**(**coneheadeat**,** 200**,** **true);**

playEating**();**

**}**

**}** **else** **if** **(**hp **>** 166**)** **{**

**if** **(!**isEating**())** **{**

animate**(**coneheadwalkd**,** 350**,** **true);**

move**(-**walkSpeed**);**

**}** **else** **{**

animate**(**coneheadeatd**,** 200**,** **true);**

playEating**();**

**}**

**}** **else** **if** **(**hp **>** 100**)** **{**

**if** **(!**isEating**())** **{**

animate**(**coneheadwalkdd**,** 350**,** **true);**

move**(-**walkSpeed**);**

**}** **else** **{**

animate**(**coneheadeatdd**,** 200**,** **true);**

playEating**();**

**}**

**}** **else** **{**

**if** **(**cone**)** **{**

cone **=** **false;**

MyWorld**.**addObject**(new** Cone**(),** getX**(),** getY**()-**25**);**

**}**

**if** **(**hp **>** 50**)** **{**

**if** **(!**isEating**())** **{**

animate**(**walk**,** 350**,** **true);**

move**(-**walkSpeed**);**

**}** **else** **{**

animate**(**eat**,** 200**,** **true);**

playEating**();**

**}**

**}** **else** **{**

**if** **(!**fallen**)** **{**

fallen **=** **true;**

Audio**.**play**(**80**,** "limbs\_pop.mp3"**);**

MyWorld**.**addObject**(new** Arm**(),** getX**()+**8**,** getY**()+**20**);**

**}**

**if** **(!**isEating**())** **{**

animate**(**armless**,** 350**,** **true);**

move**(-**walkSpeed**);**

**}** **else** **{**

animate**(**armlesseat**,** 200**,** **true);**

playEating**();**

**}**

**}**

**}**

**}**

public void hit**(**int dmg**)** **{**

Audio**.**play**(**70**,** "plastichit.mp3"**,** "plastichit2.mp3"**);**

Audio**.**play**(**70**,** "splat.mp3"**,** "splat2.mp3"**,** "splat3.mp3"**);**

**if** **(**isLiving**())** **{**

**if** **(**hp **>** 232**)** **{**

**if** **(!**isEating**())** **{**

hitFlash**(**coneheadwalk**,** "coneheadwalk"**);**

**}** **else** **{**

hitFlash**(**coneheadeat**,** "coneheadeat"**);**

**}**

**}** **else** **if** **(**hp **>** 166**)** **{**

**if** **(!**isEating**())** **{**

hitFlash**(**coneheadwalkd**,** "coneheadwalkd"**);**

**}** **else** **{**

hitFlash**(**coneheadeatd**,** "coneheadeatd"**);**

**}**

**}** **else** **if** **(**hp **>** 100**)** **{**

**if** **(!**isEating**())** **{**

hitFlash**(**coneheadwalkdd**,** "coneheadwalkdd"**);**

**}** **else** **{**

hitFlash**(**coneheadeatdd**,** "coneheadeatdd"**);**

**}**

**}** **else** **{**

**if** **(!**fallen**)** **{**

**if** **(!**eating**)** **{**

hitFlash**(**walk**,** "zombiewalk"**);**

**}** **else** **{**

hitFlash**(**eat**,** "zombieeating"**);**

**}**

**}** **else** **{**

**if** **(!**eating**)** **{**

hitFlash**(**armless**,** "armlesszombie"**);**

**}** **else** **{**

hitFlash**(**armlesseat**,** "armlesszombieeating"**);**

**}**

**}**

**}**

hp **-=** dmg**;**

**}** **else** **if** **(!**finalDeath**)** **{**

**if** **(!**eating**)** **{**

hitFlash**(**headless**,** "zombieheadless"**);**

**}** **else** **{**

hitFlash**(**headlesseating**,** "headlesszombieeating"**);**

**}**

**}**

**}**

**}**

###### 2.2.3.2.11.2.10 fallingZombie

**import** greenfoot**.\*;** // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

public class fallingZombie **extends** animatedObjects

**{**

public GreenfootImage**[]** fall**;**

public fallingZombie**(**GreenfootImage**[]** fall**)** **{**

**this.**fall **=** fall**;**

**}**

public void act**()**

**{**

**if** **(**getImage**().**getTransparency**()** **<=** 0**)** **{**

getWorld**().**removeObject**(this);**

**return;**

**}**

**if** **(**frame **<=** 7**)** **{**

animate**(**fall**,** 200**,** **false);**

**}** **else** **{**

**if** **(**getImage**().**getTransparency**()-**3 **<=** 0**)** **{**

getImage**().**setTransparency**(**0**);**

**}** **else** **{**

getImage**().**setTransparency**(**getImage**().**getTransparency**()-**3**);**

**}**

**}**

// Add your action code here.

**}**

**}**

##### 2.2.3.2.11.3 useShovel

**import** greenfoot**.\*;** // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

public class useShovel **extends** SmoothMover

**{**

MyWorld MyWorld**;**

Plant lastPlant **=** **null;**

public void addedToWorld**(**World world**)** **{**

MyWorld **=** **(**MyWorld**)**getWorld**();**

**}**

public void act**()**

**{**

MouseInfo mouse **=** Greenfoot**.**getMouseInfo**();**

**if** **(**mouse **!=** **null)** **{**

setLocation**(**mouse**.**getX**(),** mouse**.**getY**());**

**if** **(**mouse**.**getX**()** **<** SeedBank**.**x1 **||** mouse**.**getX**()** **>** SeedBank**.**x2 **||** mouse**.**getY**()** **<** SeedBank**.**y1 **||** mouse**.**getY**()** **>** SeedBank**.**y2**)** **{**

**if** **(**lastPlant **!=** **null)** **{**

lastPlant**.**opaque **=** **false;**

**}**

**if** **(**Greenfoot**.**mouseClicked**(null))** **{**

MyWorld**.**shovel**.**setSelected**(false);**

MyWorld**.**removeObject**(this);**

**return;**

**}**

**}** **else** **{**

int x **=** **(**int**)((**mouse**.**getX**()-**SeedBank**.**x1**)/**SeedBank**.**xSpacing**);**

int y **=** **(**int**)((**mouse**.**getY**()-**SeedBank**.**y1**)/**SeedBank**.**ySpacing**);**

Plant current **=** MyWorld**.**grid**.**getPlant**(**x**,** y**);**

**if** **(**current **!=** **null)** **{**

**if** **(**lastPlant **!=** **null** **&&** lastPlant **!=** current**)** **{**

lastPlant**.**opaque **=** **false;**

lastPlant **=** current**;**

**}** **else** **{**

lastPlant **=** current**;**

lastPlant**.**opaque **=** **true;**

**}**

**}** **else** **{**

**if** **(**lastPlant **!=** **null)** **{**

lastPlant**.**opaque **=** **false;**

**}**

**}**

**if** **(**Greenfoot**.**mouseClicked**(null))** **{**

**if** **(**current **!=** **null)** **{**

MyWorld**.**grid**.**removePlant**(**current**.**getXPos**(),** current**.**getYPos**());**

**}** **else** **{**

Audio**.**play**(**80**,** "tap.mp3"**,** "tap2.mp3"**);**

**}**

MyWorld**.**shovel**.**setSelected**(false);**

MyWorld**.**removeObject**(this);**

**return;**

**}**

**}**

**}**

**}**

**}**

#### 2.2.3.2.12 SunCounter

**import** greenfoot**.\*;** // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

public class SunCounter **extends** Actor**{**

public static final int x **=** 125**;**

public static final int y **=** 52**;**

public int sun **=** 50**;**

public static final int textY **=** 47**;**

private long currentFrame **=** System**.**nanoTime**();**

private long lastFrame **=** System**.**nanoTime**();**

private long deltaTime**;**

private MyWorld myWorld**;**

//sun from sky manager

public void act**()** **{**

currentFrame **=** System**.**nanoTime**();**

deltaTime **=** **(**currentFrame **-** lastFrame**)/**1000000**;**

**if** **(**deltaTime **>=** 10000L**)** **{**

myWorld**.**addObject**(new** FallingSun**(),** RNG**.**Int**(**SeedBank**.**x1**,** SeedBank**.**x2**),** 0**);**

lastFrame **=** System**.**nanoTime**();**

**}**

**}**

//start

@Override

public void addedToWorld**(**World world**)** **{**

myWorld **=** **(**MyWorld**)** getWorld**();**

currentFrame **=** System**.**nanoTime**();**

lastFrame **=** System**.**nanoTime**();**

updateText**();**

**}**

//update number

public void updateText**()** **{**

String number **=** "" **+** sun**;**

char**[]** text **=** number**.**toCharArray**();**

getImage**().**clear**();**

setImage**(**"suncounter.png"**);**

int baseX **=** 44**;**

int spacing **=** 12**;**

**if** **(**text**.**length **>** 5**)** **{**

sun **=** 99999**;**

System**.**out**.**println**(**"It seems like you don't have the patience to play our game, ok then"**);**

**}** **else** **{**

**for** **(**int i **=** 0**;** i **<** text**.**length**;** i**++)** **{**

int offsetX **=** **switch** **(**text**.**length**)** **{**

**case** 5 **->** 20 **+** i **\*** spacing**;**

**case** 4 **->** 26 **+** i **\*** spacing**;**

**case** 3 **->** 33 **+** i **\*** spacing**;**

**case** 2 **->** 38 **+** i **\*** spacing**;**

**default** **->** baseX**;**

**};**

getImage**().**drawImage**(new** GreenfootImage**(**"text" **+** text**[**i**]** **+** ".png"**),** offsetX**,** textY**);**

**}**

**}**

**}**

//add

public void addSun**(**int amount**)** **{**

sun **+=** amount**;**

updateText**();**

**}**

//remove

public void removeSun**(**int amount**)** **{**

sun **-=** amount**;**

updateText**();**

**}**

**}**

#### 2.2.3.2.13 Transistion

**import** greenfoot**.\*;** // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

/\*\*

\* Write a description of class Transition here.

\*

\* **@author** (your name)

\* **@version** (a version number or a date)

\*/

public class Transition **extends** Actor

**{**

/\*\*

\* Act - do whatever the Transition wants to do. This method is called whenever

\* the 'Act' or 'Run' button gets pressed in the environment.

\*/

private World targetWorld**;**

private boolean fadeOut**;**

private int fadeSpeed**;**

private int a **=** 0**;**

private int counter **=** 0**;**

private int waitDuration **=** 0**;**

public Transition**(**boolean fadeOut**,** World targetWorld**,** int fadeSpeed**)** **{**

**this.**fadeOut **=** fadeOut**;**

**this.**targetWorld **=** targetWorld**;**

**this.**fadeSpeed **=** fadeSpeed**;**

getImage**().**setTransparency**(**0**);**

**}**

public Transition**(**boolean fadeOut**,** World targetWorld**,** int fadeSpeed**,** int waitDuration**)** **{**

**this(**fadeOut**,** targetWorld**,** fadeSpeed**);**

**this.**waitDuration **=** waitDuration**;**

**}**

public Transition**(**boolean fadeOut**,** World targetWorld**,** String imageFile**,** int fadeSpeed**)** **{**

**this.**fadeOut **=** fadeOut**;**

**this.**targetWorld **=** targetWorld**;**

**this.**fadeSpeed **=** fadeSpeed**;**

setImage**(new** GreenfootImage**(**imageFile**));**

getImage**().**setTransparency**(**0**);**

**}**

public void act**()** **{**

a **+=** fadeSpeed**;**

**if** **(**a **<=** 255**)** **{**

getImage**().**setTransparency**(**a**);**

**}** **else** **{**

getImage**().**setTransparency**(**255**);**

counter**++;**

**if** **(**counter **>** waitDuration**)** **{**

**if** **(**fadeOut**)** **{**

getWorld**().**addObject**(new** EndTransition**(),** 329**,** 224**);**

**}**

changeWorld**();**

**return;**

**}**

**}**

**}**

private void changeWorld**()** **{**

getWorld**().**removeObject**(this);**

Greenfoot**.**setWorld**(**targetWorld**);**

**}**

public void addedToWorld**(**World world**)** **{**

**}**

**}**

#### 2.2.3.2.14 TransparentObject

**import** greenfoot**.\*;** // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

/\*\*

\* Write a description of class TransparentObject here.

\*

\* **@author** (your name)

\* **@version** (a version number or a date)

\*/

public class TransparentObject **extends** Actor**{**

public TransparentObject**(){**

**}**

public void act**(){**

**}**

public void setTransparent**(**boolean bool**){**

**if** **(**bool**)** **{**

getImage**().**setTransparency**(**125**);**

**}** **else** **{**

getImage**().**setTransparency**(**255**);**

**}**

**}**

**}**

##### 2.2.3.2.14.1 TransparentCactus

**import** greenfoot**.\*;** // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

public class TransparentCactus **extends** TransparentObject

**{**

public TransparentCactus**(**boolean bool**)** **{**

**if** **(**bool**)** **{**

getImage**().**setTransparency**(**125**);**

**}** **else** **{**

getImage**().**setTransparency**(**255**);**

**}**

**}**

public void act**()**

**{**

// Add your action code here.

**}**

**}**

##### 2.2.3.2.14.2 TransparentPeashooter

**import** greenfoot**.\*;** // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

public class TransparentPeashooter **extends** TransparentObject

**{**

public TransparentPeashooter**(**boolean bool**)** **{**

**if** **(**bool**)** **{**

getImage**().**setTransparency**(**125**);**

**}** **else** **{**

getImage**().**setTransparency**(**255**);**

**}**

**}**

public void act**()**

**{**

// Add your action code here.

**}**

**}**

##### 2.2.3.2.14.3 TransparentPotato

**import** greenfoot**.\*;** // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

public class TransparentPotato **extends** TransparentObject

**{**

public TransparentPotato**(**boolean bool**)** **{**

**if** **(**bool**)** **{**

getImage**().**setTransparency**(**125**);**

**}** **else** **{**

getImage**().**setTransparency**(**255**);**

**}**

**}**

public void act**()**

**{**

// Add your action code here.

**}**

**}**

##### 2.2.3.2.14.4 TransparentRepeater

**import** greenfoot**.\*;** // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

public class TransparentRepeater **extends** TransparentObject

**{**

public TransparentRepeater**(**boolean bool**)** **{**

**if** **(**bool**)** **{**

getImage**().**setTransparency**(**125**);**

**}** **else** **{**

getImage**().**setTransparency**(**255**);**

**}**

**}**

public void act**()**

**{**

// Add your action code here.

**}**

**}**

##### 2.2.3.2.14.5 TransparentSunflower

**import** greenfoot**.\*;** // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

public class TransparentSunflower **extends** TransparentObject

**{**

public TransparentSunflower**(**boolean bool**)** **{**

**if** **(**bool**)** **{**

getImage**().**setTransparency**(**125**);**

**}** **else** **{**

getImage**().**setTransparency**(**255**);**

**}**

**}**

public void act**()**

**{**

// Add your action code here.

**}**

**}**

##### 2.2.3.2.14.6 TransparentWallnut

**import** greenfoot**.\*;** // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

public class TransparentWalnut **extends** TransparentObject

**{**

public TransparentWalnut**(**boolean bool**)** **{**

**if** **(**bool**)** **{**

getImage**().**setTransparency**(**125**);**

**}** **else** **{**

getImage**().**setTransparency**(**255**);**

**}**

**}**

public void act**()**

**{**

// Add your action code here.

**}**

**}**

#### 2.2.3.2.15 WaveManager

**import** greenfoot**.\*;** // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

**import** java**.**util**.\*;**

/\*\*

\* Write a description of class WaveManager here.

\*

\* **@author** (your name)

\* **@version** (a version number or a date)

\*/

public class WaveManager **extends** Actor

**{**

public long currentFrame **=** System**.**nanoTime**();**

public static final int xOffset **=** 760**;**

public static final int yOffset **=** 65**;**

public static final int ySpacing **=** 73**;**

public ArrayList**<**ArrayList**<**Zombie**>>** zombieRow **=** **new** ArrayList**<**ArrayList**<**Zombie**>>();**

public ArrayList**<**Zombie**>** row1 **=** **new** ArrayList**<**Zombie**>();**

public ArrayList**<**Zombie**>** row2 **=** **new** ArrayList**<**Zombie**>();**

public ArrayList**<**Zombie**>** row3 **=** **new** ArrayList**<**Zombie**>();**

public ArrayList**<**Zombie**>** row4 **=** **new** ArrayList**<**Zombie**>();**

public ArrayList**<**Zombie**>** row5 **=** **new** ArrayList**<**Zombie**>();**

public long lastFrame **=** System**.**nanoTime**();**

public Zombie**[][]** level**;**

public long levelTime**;**

public long waveTime**;**

public long firstWave**;**

public long deltaTime**;**

public long deltaTime2**;**

public boolean won **=** **false;**

public World MyWorld**;**

public int wave **=** **-**1**;**

public boolean first **=** **false;**

public boolean finishedSending **=** **false;**

public int**[]** hugeWaves**;**

public WaveManager**(**long timeBetweenWaves**,** Zombie**[][]** level**,** long firstWave**,** boolean first**,** int**...** hugeWaves**)** **{**

**this.**level **=** level**;**

**this.**levelTime **=** levelTime**;**

**this.**waveTime **=** timeBetweenWaves**;**

**this.**firstWave **=** firstWave**;**

**this.**hugeWaves **=** hugeWaves**;**

**this.**first **=** first**;**

zombieRow**.**add**(**row1**);**

zombieRow**.**add**(**row2**);**

zombieRow**.**add**(**row3**);**

zombieRow**.**add**(**row4**);**

zombieRow**.**add**(**row5**);**

**}**

public void startLevel**()** **{**

wave **=** 0**;**

Audio**.**play**(**80**,** "readysetplant.mp3"**);**

MyWorld**.**addObject**(new** ReadySetPlant**(),** 400**,** 230**);**

**}**

//Fix order cause no setPaintOrder for actors :(

public void fixOrder**()** **{**

List**<**Zombie**>** zombies **=** MyWorld**.**getObjects**(**Zombie**.**class**);**

**for** **(**int r **=** 0**;** r **<** 5**;** r**++)** **{**

**for** **(**int i **=** 0**;** i **<** zombies**.**size**();** i**++)** **{**

**if** **(**zombies**.**get**(**i**).**getWorld**()** **!=** **null** **&&** zombies**.**get**(**i**).**getYPos**()** **==** r**)** **{**

int x **=** zombies**.**get**(**i**).**getX**();**

int y **=** zombies**.**get**(**i**).**getY**();**

**try** **{**

MyWorld**.**removeObject**(**zombies**.**get**(**i**));**

MyWorld**.**addObject**(**zombies**.**get**(**i**),** x**,** y**);**

**}** **catch** **(**Exception ex**)** **{**

System**.**out**.**println**(**"Fix Order Error"**);**

**}**

**}**

**}**

**}**

**}**

public void act**()**

**{**

**if** **(**wave **!=** **-**1**)** **{**

currentFrame **=** System**.**nanoTime**();**

deltaTime **=** **(**currentFrame **-** lastFrame**)** **/** 1000000**;**

**}** **else** **{**

lastFrame **=** System**.**nanoTime**();**

**}**

**if** **(**wave **>** level**.**length**-**1**)** **{**

MyWorld**.**addObject**(new** finishedSending**(this,** 15000L**),** 0**,**0**);**

wave **=** **-**1**;**

**}**

**if** **(**deltaTime **>=** firstWave **&&** wave **!=** **-**1 **&&** first **==** **true)** **{**

Audio**.**play**(**80**,** "awooga.mp3"**);**

checkSendWave**();**

wave**++;**

lastFrame **=** System**.**nanoTime**();**

first **=false;**

**}**

**if** **(**first **==** **false** **&&** wave **!=** **-**1**)** **{**

**if** **((**deltaTime **>=** waveTime**)** **||** MyWorld**.**getObjects**(**Zombie**.**class**).**size**()** **==** 0**)** **{**

checkSendWave**();**

wave**++;**

lastFrame **=** System**.**nanoTime**();**

**}**

**}**

**}**

public void checkSendWave**()** **{**

**for** **(**int i **:** hugeWaves**)** **{**

**if** **(**i **==** wave**)** **{**

**if** **(**wave **==** level**.**length**-**1**)** **{**

Audio**.**play**(**70**,** "hugewave.mp3"**);**

finishedSending **=** **false;**

sendHugeWave**(**level**[**wave**]);**

MyWorld**.**addObject**(new** AHugeWave**(true),**360**,**215**);**

**return;**

**}** **else** **{**

Audio**.**play**(**70**,** "hugewave.mp3"**);**

finishedSending **=** **false;**

sendHugeWave**(**level**[**wave**]);**

MyWorld**.**addObject**(new** AHugeWave**(false),**360**,**215**);**

**return;**

**}**

**}**

**}**

sendWave**(**level**[**wave**]);**

**}**

@Override

protected void addedToWorld**(**World world**)** **{**

MyWorld **=** **(**MyWorld**)**getWorld**();**

lastFrame **=** System**.**nanoTime**();**

currentFrame **=** System**.**nanoTime**();**

ProgressionBar progressBar **=** **new** ProgressionBar**(this);**

world**.**addObject**(**progressBar**,** 490**,** 25**);**

**}**

public boolean hasWon**()** **{**

**if** **(**wave **==** **-**1 **&&** finishedSending **&&** MyWorld**.**getObjects**(**Zombie**.**class**).**size**()** **==** 0**)** **{**

won **=** **true;**

**}** **else** **{**

won **=** **false;**

**}**

**return** won**;**

**}**

public void sendWave**(**Zombie**[]** wave**)** **{**

**for** **(**int i **=** 0**;** i **<** wave**.**length**;** i**++)** **{**

**if** **(**i **<** 5**)** **{**

**if** **(**wave**[**i**]!=null)** **{**

//Send!

MyWorld**.**addObject**(**wave**[**i**],** xOffset**,** **(**i**%**5**)\***ySpacing**+**yOffset**);**

zombieRow**.**get**(**i**%**5**).**add**(**wave**[**i**]);**

**}**

**}** **else** **{**

//If more then 1 zombie per row, delay depending on how many

**if** **(**wave**[**i**]** **!=** **null)** **{**

finishedSending **=** **false;**

int wait **=** **(**int**)(**i**/**5**);**

int offset **=** xOffset**+**wait**\***20**;**

MyWorld**.**addObject**(**wave**[**i**],** offset**,** **(**i**%**5**)\***ySpacing**+**yOffset**);**

zombieRow**.**get**(**i**%**5**).**add**(**wave**[**i**]);**

**}**

**}**

**}**

MyWorld**.**addObject**(new** FixOrder**(this,** 1000L**),** 0**,**0**);**

**}**

public void sendHugeWave**(**Zombie**[]** wave**)** **{**

**for** **(**int i **=** 0**;** i **<** wave**.**length**;** i**++)** **{**

**if** **(**i **<** 5**)** **{**

**if** **(**wave**[**i**]!=null)** **{**

//Send

MyWorld**.**addObject**(**wave**[**i**],** xOffset**+**50**,** **(**i**%**5**)\***ySpacing**+**yOffset**);**

zombieRow**.**get**(**i**%**5**).**add**(**wave**[**i**]);**

**}**

**}** **else** **{**

//If more then 1 zombie per row, delay depending on how many

**if** **(**wave**[**i**]** **!=** **null)** **{**

finishedSending **=** **false;**

int wait **=** **(**int**)(**i**/**5**);**

int offset **=** xOffset**+**50**+**wait**\***20**;**

MyWorld**.**addObject**(**wave**[**i**],** offset**,** **(**i**%**5**)\***ySpacing**+**yOffset**);**

zombieRow**.**get**(**i**%**5**).**add**(**wave**[**i**]);**

**}**

**}**

**}**

MyWorld**.**addObject**(new** FixOrder**(this,** 1000L**),** 0**,**0**);**

**}**

**}**

#### 2.2.3.2.16 finishedSending

**import** greenfoot**.\*;**

class finishedSending **extends** Actor**{**

public long deltaTime**;**

public long lastFrame **=** System**.**nanoTime**();**

public long currentFrame **=** System**.**nanoTime**();**

public long delayTime**;**

public WaveManager level**;**

finishedSending**(**WaveManager level**,** long delayTime**)** **{**

**this.**delayTime **=** delayTime**;**

**this.**level **=** level**;**

**}**

public void act**()** **{**

currentFrame **=** System**.**nanoTime**();**

deltaTime **=** **(**currentFrame **-** lastFrame**)** **/** 1000000**;**

**if** **(**deltaTime **>** delayTime**)** **{**

level**.**finishedSending **=** **true;**

getWorld**().**removeObject**(this);**

**return;**

**}**

**}**

**}**

### 2.2.3.3 Extensions

#### 2.2.3.3.1 Audio

**import** greenfoot**.**GreenfootSound**;**

**import** greenfoot**.**Greenfoot**;**

public class Audio**{**

public Audio**(){**

**}**

public static void play**(**String**...** audio**){**

**if** **(**audio **==** **null** **||** audio**.**length **==** 0**)** **return;**

int index **=** Greenfoot**.**getRandomNumber**(**audio**.**length**);**

GreenfootSound sound **=** **new** GreenfootSound**(**audio**[**index**]);**

sound**.**play**();**

**}**

public static void play**(**int volume**,** String**...** audio**){**

**if** **(**audio **==** **null** **||** audio**.**length **==** 0**)** **return;**

int index **=** Greenfoot**.**getRandomNumber**(**audio**.**length**);**

GreenfootSound sound **=** **new** GreenfootSound**(**audio**[**index**]);**

sound**.**setVolume**(**volume**);**

sound**.**play**();**

**}**

public static void play**(**boolean loop**,** String**...** audio**){**

**if** **(**audio **==** **null** **||** audio**.**length **==** 0**)** **return;**

int index **=** Greenfoot**.**getRandomNumber**(**audio**.**length**);**

GreenfootSound sound **=** **new** GreenfootSound**(**audio**[**index**]);**

**if** **(**loop**){**

sound**.**playLoop**();**

**}else{**

sound**.**play**();**

**}**

**}**

public static void stop**(**GreenfootSound**...** sounds**){**

**for** **(**GreenfootSound sound **:** sounds**){**

**if** **(**sound **!=** **null){**

sound**.**stop**();**

**}**

**}**

**}**

**}**

#### 2.2.3.3.2 RNG

/\*\*

\* Write a description of class RNG here.

\*

\* **@author** (your name)

\* **@version** (a version number or a date)

\*/

public class RNG**{**

private int x**;**

public RNG**(){**

**}**

public static int Int**(**int min**,** int max**)** **{**

**return** **(**int**)((**Math**.**random**()** **\*** **(**max**-**min**))** **+** min**);**

**}**

public static double Double**(**int min**,** int max**)** **{**

**return** **((**Math**.**random**()** **\*** **(**max**-**min**))** **+** min**)/**100**;**

**}**

**}**

#### 2.2.3.3.3 Timer1

**import** java**.**util**.**TimerTask**;**

**import** greenfoot**.\*;**

/\*\*

\* Write a description of class Timer here.

\*

\* **@author** (your name)

\* **@version** (a version number or a date)

\*/

class Timer1 **extends** TimerTask **{**

private GreenfootImage**[]** imageArray**;**

private String spriteSheet**;**

private int frameIndex1**;**

private int frameIndex2**;**

public Timer1**(**GreenfootImage**[]** imageArray**,** String filename**,** int frameIndex1**,** int frameIndex2**){**

**if** **(**imageArray **==** **null** **||** filename **==** **null** **||** filename**.**isEmpty**())** **{**

**throw** **new** IllegalArgumentException**(**"Invalid"**);**

**}**

**this.**imageArray **=** imageArray**;**

**this.**spriteSheet **=** filename**;**

**this.**frameIndex1 **=** frameIndex1**;**

**this.**frameIndex2 **=** frameIndex2**;**

**}**

@Override

public void run**(){**

**if** **(**frameIndex1 **>=** 0 **&&** frameIndex1 **<** imageArray**.**length**)** **{**

imageArray**[**frameIndex1**]** **=** **new** GreenfootImage**(**spriteSheet **+** **(**frameIndex1 **+** 1**)** **+** ".png"**);**

**}**

**if** **(**frameIndex2 **>=** 0 **&&** frameIndex2 **<** imageArray**.**length**)** **{**

imageArray**[**frameIndex2**]** **=** **new** GreenfootImage**(**spriteSheet **+** **(**frameIndex2 **+** 1**)** **+** ".png"**);**

**}**

**}**

**}**

# SUMMARY

## 3.1 Project Recap:

– *Plants vs Zombie* is a tower-defense game developed as an Object-Oriented Programming (OOP) assignment.

– The project was created by a team of five first-year students at International University, merging classic gameplay with our academic and collaborative learning experiences.

## 3.2 Key Goals:

– Enhance our understanding of OOP principles through hands-on coding and design.

– Develop teamwork, effective communication, and collaborative coding skills.

– Create a functional and engaging game that challenges players to strategically place plants to defeat invading zombies.

## 3.3 Methodology and Tools:

**–** Programming & Environment:

* Developed using **Java** and the **Greenfoot** platform, which provides an interactive simulation environment.

**–** Design & Documentation:

* Utilized **Mindmeiser** to generate the initial UML mindmap, aiding in conceptualizing the project structure.
* Employed **Lucidchart** for creating detailed UML diagrams that outline the system architecture.

**–** Collaboration:

* Managed version control and collaborative development through **GitHub**.

## 3.4 Outcomes and Achievements:

– Successfully implemented key gameplay elements reflecting OOP concepts such as modularity, inheritance, and encapsulation.

– Strengthened both individual coding proficiency and team collaboration skills.

– Gained valuable insights into the iterative process of game development, from design brainstorming to code execution.

# REFERENCES

– Plants vs Zombies by rohangoel96: <https://github.com/rohangoel96/PlantsVsZombies-Game>

– Plants vs Zombies by TheExploration: <https://github.com/TheExploration/Plants-Vs-Zombies>

– Watch our Gameplay Demo: <https://www.youtube.com/watch?v=vKbh2-2IUVM>