|  |
| --- |
| **Zombies** |
| private Humans target;  private int hungerLevel;  private int dmg;  private int attactCounter;  private int hungerCounter;  Has the ability to wander around, attack humans to convert them/eat them, and use basic functionality of Char class. |

|  |
| --- |
| **Char** |
| protected int health;  protected int speed;  protected ZombieWorld myWorld;  This abstract class has the functionality to wander around, avoid obstacles, and turn once it reaches an edge of the world. |

|  |
| --- |
| **Soldiers** |
| private int shotDelay;  private int rateOfFire;  private int bulletSpeed;  private int shootRange;  private int searchRange;  private int numBullets;  private int outCounter;  private Ammo ammo;  Has the ability to shoot at zombies when they are in range, have an icon for ammo if it can shoot, and use all basic human functionality |

|  |
| --- |
| **Humans** |
| protected int baseId;  protected boolean target ;  protected int target;  protected int targetY;  protected int foodCarry;  protected int weaponCarry;  protected ZombieWorld myWorld;  This abstract class has the functionality to take damage, convert to zombies, determine where they are trying to go, and return to base. |

|  |
| --- |
| **Scouts** |
| private boolean priFood;  private int getSupCounter;  private boolean carryingSup;  private boolean gettingSup;  private boolean goFromTop;  protected Food food;  protected Weapon weapon;  Has the ability to determine what supplies to get, pick up supplies, have an icon for what it picks up, and use all basic human functionality. |

|  |
| --- |
| **Obstacle** |
| Abstract class used to categorize obstacles. |

|  |
| --- |
| **Rock** |
| One type of obstacle that will have to be avoided |

|  |
| --- |
| **Tree** |
| Another type of obstacle that will have to be avoided |

|  |
| --- |
| **Food** |
| One type of icon that represents the carrying of food by a scout. |

|  |
| --- |
| **Rock** |
| One type of obstacle that will have to be avoided |

|  |
| --- |
| **Icons** |
| protected Humans parent;  Class used to categorize icon types. The icons will follow a child of the human class (scout or soldier) |

|  |
| --- |
| **Weapon** |
| One type of icon that represents the carrying of weapons by a scout. |

|  |
| --- |
| **Ammo** |
| One type of icon that represents ammo available for soldiers to shoot. |

|  |
| --- |
| **Base** |
| private int baseId;  private int height;  private int width;  private int randSpawnRate;  private int spawnXDefault;  private int spawnYDefault;  private int spawnX;  private int spawnY;  private int population;  private int bulletSpeed;  private int rateOfFire;  private int shootRange;  private Turret turret;  private Statistics statBar;  private int statBarX;  private int statBarY;  private List<Supplies> supplies;  private List<Zombie> totalZombies;  private int consumptionDelay;  private int consumptionRate;  private int deathDelay;  private int deathRate;  private int shotDelay;  private int populateCounter;  private int populateRate;  private int massSpawnDelay;  This class controls a majority of the simulation. This includes spawning humans, tracking base supplies and population, using a turret to shoot objects, and having stat bars to display the situation inside each base. Uses functionality of structure. |

|  |
| --- |
| **Structure** |
| protected int weapons;  protected int food;  This Abstract class allows structures to determine how much food and weapons they possess. |

|  |
| --- |
| **Supplies** |
| This class includes supply creates for food and weapons, and destroys crates upon depletion of supplies.. It uses the functionality of structure. |

|  |
| --- |
| **Bullet** |
| This class has the ability to deal damage when it hits a zombie, rotate towards target, and uses the basic functionality of the projectile class. |

|  |
| --- |
| **Projectile** |
| protected int width = getImage().getWidth();  protected int height = getImage().getHeight();  protected int Vx;  protected int Vy;  protected Zombie target;  This abstract class contains functionality of both bullet types. Such as hitting objects, and what happens if they go out of bounds |

|  |
| --- |
| **PlaneBullet** |
| This class has the ability to deal damage when it hits a zombie, will be removed if it goes off screen, and uses the basic functionality of the projectile class. Very similar to bullet but it is fired from turrets on top of a plane, meaning different flight paths, targeting, etc… |

|  |
| --- |
| **SupplySpawning** |
| Class simply used to category all things related to supply drop. |

|  |
| --- |
| **ParachuteMarker** |
| private ZombieWorld myWorld;  Marker for checking whether supply crate dropped will touch an obstacle or not. If so it will not be added. |

|  |
| --- |
| **Plane** |
| private int speed;  private int turretSpeed;  private int parachuteNum;  private int shotDelay;  private int shootRange;  private int rateOfFire;  private int bulletSpeed;  private Turret turret1;  private Turret turret2;  Will fly across the map shooting bullets at zombies via turrets. There will be two turrets on both sides of planes. |

|  |
| --- |
| **Parachute** |
| private int width = getImage().getWidth();  private int height = getImage().getHeight();  private int counter;  Parachute will shrink until a certain size and then will be removed and drops a supply crate. Uses |

|  |
| --- |
| **DisplayBars** |
| Abstract class that helps with displaying all info on the screen. Has the main functionality to make the bars visible. |

|  |
| --- |
| **Statistics** |
| private int population;  private int food;  private int weapons;  private Color backgroundColor;  private Color textColorPop;  private Color textColorFood;  private Color textColorWeap;  private Font textFont;  private GreenfootImage stats;  private GreenfootImage background  private GreenfootImage foodIcon;  private GreenfootImage weaponsIcon;  private GreenfootImage populationIcon;  private GreenfootImage foodValue;  private GreenfootImage weaponsValue;  private GreenfootImage populationValue;  The statistics class displays the values for each specific base. Uses basic functionality of DisplayBars class. Additionally it has small icons for food, weapons, and population, |

|  |
| --- |
| **StatBar** |
| private GreenfootImage bg;  private GreenfootImage fg;  private GreenfootImage img;  private ZombieWorld myWorld;  Keeps track of statistics for character currently on screen. This includes scouts, zombies, and soldiers. Uses basic functionality of DisplayBars class. When clicked it will go from half-transparent to fully visible. |

|  |
| --- |
| **Turret** |
| private int width = getImage().getWidth();  private int height = getImage().getHeight();  private int shootRange;  private List<Zombie> targets;  Turret is a class that is not under a larger subcategory. It determines the angles necessary and rotation to shoot at zombie targets. It will be atop all four bases. |

|  |
| --- |
| **Buttons** |
| protected boolean clicked;  Will be used so that when a button is clicked certain actions will be performed. |

|  |
| --- |
| **StartScreenButton** |
| private GreenfootImage button;  private GreenfootImage buttonHover;  private StartScreen myWorld;  The class is used so that when the start button is pressed the start screen will transition towards the actual simulation. Uses functionality of Buttons class. |

|  |
| --- |
| **TransitionFade** |
| GreenfootSound theme;  GreenfootImage myImage;  FastImage image;  int[] color = {255,255,255,255};  MouseInfo mouse;  int mouseX,mouseY;  int prevX,prevY;  boolean pressed = false;  int type=1;  int redCycle=600,redVel=1;  int greenCycle=400,greenVel=-1;  int blueCycle=200,blueVel=1;  private int counter = 0;  private boolean exists = true;  The class is used so that after the start button is pressed a smooth transition is made between that screen and the actual simulation. Uses GreenFoot World functionality. |

|  |
| --- |
| **World** |
| GreenFoot class that is generically used to run simulations or programs. This is a super class in a way for all the worlds within this simulation. |

|  |
| --- |
| **LoadingScreen** |
| private int counter = 0;  private boolean exists = true;  private GreenfootSound theme;  Loading screen used to delay before the simulation is run. Uses GreenFoot World functionality. |

|  |
| --- |
| **ZombieWorld** |
| private GreenfootSound theme;  private GreenfootSound fly;  private StatBar statbar;  private List <Supplies> supplies;  private int totalSupply;  private int planeSpawnCounter;  private int randomPlaneFacing;  private int zombieOnScreen;  private int zombieDie;  private int totalZombieSpawn;  private int scoutOnScreen;  private int soldierOnScreen;  private int humanKilledCount;  private int humanConvertedCount;  private int zombieKilledCount;  private Base base1;  private Base base2;  private Base base3;  private Base base4;  The class is used to maintain and control the entire simulation within. This is the place where all elements of the code come together so that a user can view it. Uses basic functionality of GreenFoot World class. Plays music throughout simulation. Keeps track of zombie deaths, human deaths, supply related tracking, etc… Additionally, this is where all objects are added onto the screen. |

|  |
| --- |
| **StartScreen** |
| private GreenfootSound theme = new GreenfootSound("The Walking Dead Theme.mp3");  Start screen that is used to initialize the transition towards the simulation. Uses GreenFoot World functionality. |