Toni Giacchi

CSci 2113: Software Engineering

Project 2: Zombie Simulator

November 20, 2016

In this project, I am creating a city which contain humans running around, avoiding zombies. A random zombie is placed at the start of the project and contaminates humans when it is adjacent to one. In order to add more zombies, the user can simply click to a location, except on buildings, into the city panel and a new zombie will be added with each click of the mouse. When the user wants to restart the simulator, he or she just needs to click the space bar. I will go through each class, explaining its need and its functions.

DotPanel

The DotPanel adds the features need to draw out the simulator. This class comes completed. It first contains a wide-variety of colors to choose from to use. Next, there are simple methods which get the dimensions of the dot. The dots in this are the pixels of the panel. Some of the dots will be humans or zombies. Then a method initiates the panel to be drawn. The next method actually paints the panel to the user's screen to be seen. There is a repaintAndSleep method for when the panel is restarted by the user. The purpose for this class is to create methods for actually drawing out the panel we will be interacting with.

Helper

The Helper class is mainly for the purpose of creating random numbers. This class also comes completed. We use this class when we find a random location for the human or zombie for x and y coordinates.

ZombieSim

ZombieSim is the actual class that is used to run our project. It extends JFrame to draw out the simulator and implements Listener interfaces to allow the user to interact with the simulator. The ZombieSim method is what is used to actually create the design of the panel with a title, size, and color. In this, we also create a world to add humans and zombies to. We also add our listeners to allow users to react via their mouse or keyboard. When implementing these interfaces, we must also implement their methods. The ones we actually use are mouseClicked and keyTyped. MouseClicked is for the MouseListener and adds a zombie to the location where the mouse is clicked. The mouse must be clicked inside the panel and not inside buildings. If not clicked in the allowed space, nothing will happen. In keyTyped, a new world is generated and created if and only if the space bar is pressed.

City

City is an important class because it is where we create the world the user sees. A height and width is given for its size. Walls are created as Booleans in order to test if there are at a certain location or not. There are also arrays for the humans and zombies because they will be existing in the city. The first method we implement is populate. This first goes through the array of humans and for each human, a new location is generated and a human is added to that location with its given coordinates. Next a new location is generated for a zombie that will be randomly placed at the beginning of the simulator. For both zombies and humans, the rule is put that if there is a wall at the generated location, the object will not be placed there and a new location will be generated. Random buildings are created at the start of each simulator and whenever the user restarts it. The size and location of them vary each time so there is a need to keep track of this with the walls array. The update method is what constants updates the state of the city in each step. This moves the zombies and humans at each second by going through their array list, getting each object's position and generating its next step by calling a move method in the object's class. The zombies and humans are finally seen in the draw method. I go through the arraylist of the humans and zombies, get their location, and paint a dot there either in green for humans or red for zombies.

Humans

The Humans class creates the human object. In the city, there are 200 humans at the start of the simulator. The humans have x and y coordinates and a direction. The move method has a human move in either four direction based on the fact that there is no wall and it is not out of bounds of the panel. The detect method checks to see where a zombie is. If a zombie is present within 10 squares (but not a square next to) of the human's direction then the human turns to the opposite direction it is facing. In the moveHuman method is where the human actually moves. If a zombie is detected, the human will move two spaces away from it. If not, the pickDir variable calculates a random number with the use of the Helper class. If the number is greater than zero, it will take a step in the direction it is facing. If pickDir equals zero, then it will change directions.

Zombies

The Zombie class is very similar to the Humans class, but has slightly different rules. It still contains the same variables: x and y coordinates and a direction. It has a move method to have it move one square if there is not a wall or end of the panel. The detect method tests to see if there is a human 10 squares within the direction the zombie is facing. In the moveZombie method, it is tested if a human is in one square of the zombie. If so, that human and turned into a zombie and is added to the zombie array list after being removed from the human arraylist. If a zombie sees a human within its range, it will take one step towards the human. If not, it will generates a direction. If the direction is less than or equal to 1, it will pick another direction. If not, it will continue in that direction.

EXTRA:

My addition to the project is simple, but useful, in my opinion. By simply pressing ‘p’, the simulator is paused for a few seconds. This is cool because you get to see the step that the simulator is at before it updates. By going into ZombieSim and going to the keyTyped method, one is able to adjust the pause to be longer or shorter.