AMRITA VISHWA VIDYAPEETHAM

Dodge the Enemy

Ruthvik.K
BL.EN.U4AIE19054
II sem C Section (CSE-AI)
19AIE111- DATA STRUCTURES AND ALGORITHMS 1

ABSTRACT

- The game is about moving the player in such a way that the enemies don’t hit the player’s character.
- The game is level based and difficulty increases with the increase in the level.
- A shop option is present that can be accessed any time during the game to purchase upgrades for the character like speed and health and to refill health.
- There are bosses that shoot bullets, enemies that follow the character slowly, fast and slow enemies and enemies that randomly move.

Classes and Modules

- Game.java: - The main class that runs the game is the Game class.
- GameObject.java: - It contains the location of the object in the game.
- Handler.java: - Data structure is used to handle objects.
- Menu.java: - Displays the details on the screen when the game is started.
- HUD.java : - Heads Up Display: - Displays the health capacity, score and level
- KeyInput.java: - The key inputs for the player.
- ID.java: - Defines the character of the game for health reduction.
- Player.java: - Player is the main character of the game. All controls of the player is in this class.

Classes and Modules

- MenuParticle.java: - Contains the background effects of the menu screen.
- Trail.java: - The trail that every character has while moving.
- Window.java: - Used to adjust the size of the game screen.
- Spawn.java: - The class that spawns enemies.
- Enemies: - There are different types of enemies such as basic enemy, fast enemy, boss, smart enemy. Each enemy has different characteristics. Basic Enemy just bounces off the walls, the smart enemy follows the player, the boss shoots out bullets, etc.
- Shop.java: -
  - Upgrade health increases the health capacity and refills the health.
  - Upgrade speed increases the speed of the player.
  - Refill health refills the health to maximum.
  - The cost of each upgrade increases on every purchase.

Data Structures used

- In built Linked list is used to initialize the character and check for the overlapping of the characters.
- It is used in the handler class to get the game object from GameObject class for the coordinates and ID of the object.

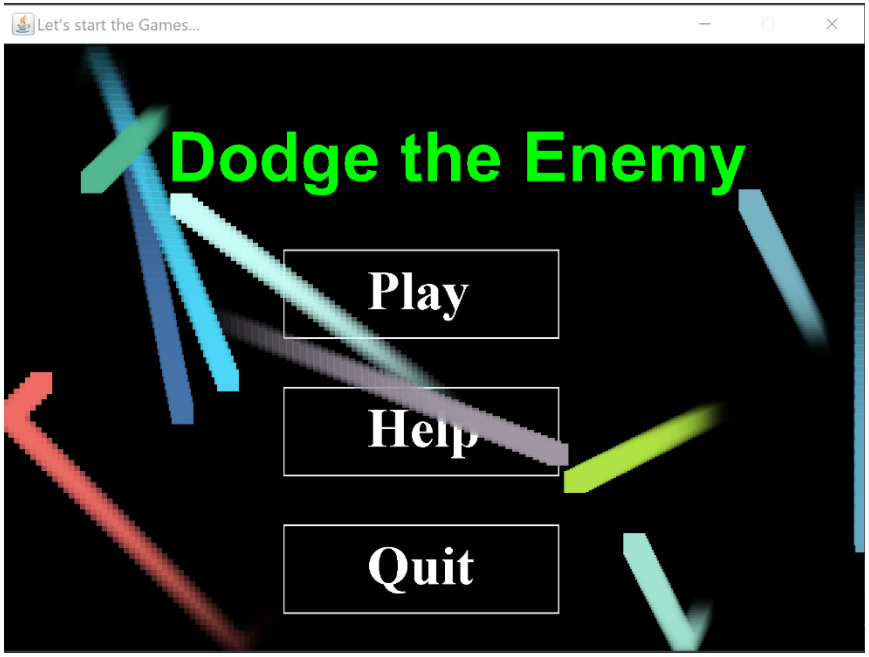
Sample code

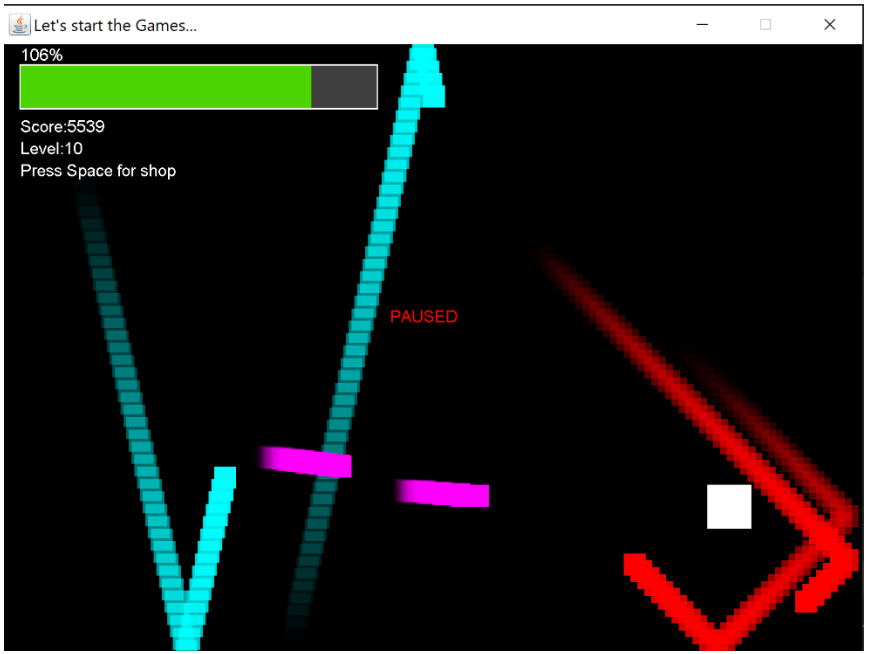
```
package Main_Project;
import java.awt.Graphics;
import java.util.LinkedList;
import Main_Project.Game.STATE;
//render all objects, controls all objects and individually update them
public class Handler {
LinkedList<GameObject> object = new LinkedList<GameObject>();
//speed of the player
public int move = 5;
//tick and render functions keeps taking in temporary objects for the value of i(ie:-each object at a time)
public void tick() {
//this loops for all the game objects present in the game
for(int i =0;i<object.size();i++)
{
GameObject tempObject = object.get(i);

tempObject.tick();
}
}
public void render(Graphics g) {
for(int i =0;i<object.size();i++)
{
GameObject tempObject = object.get(i);
tempObject.render(g);
}
}
```

```
//for removing all present enemies for the boss level
public void clearEnemies() {
for(int i =0;i<object.size();i++)
{
GameObject tempObject = object.get(i);
if(tempObject.getId()== ID.Player) {
object.clear();
addObject(new Player(tempObject.x, tempObject.y, ID.Player, this));
}
}
}
public void addObject(GameObject object) {
this.object.add(object);
}
//function used to make the boss bullets disappear
public void removeObject(GameObject object) {
this.object.remove(object);
}
}
```

Sample Output





Thank you