CS102	**Spring 2021/22**	Project Group	2C
Instructor:	**Aynur Dayanık**		
Assistant:	**Sinan Sonlu**		

Criteria	TA/Grader	Instructor
Presentation		
Overall		

~ Defender of the Galaxy~

Aynen

Mert Fidan

Tolga Han Arslan

Umut Arda Filiz

İrfan Hakan Karakoç

Detailed Design Report

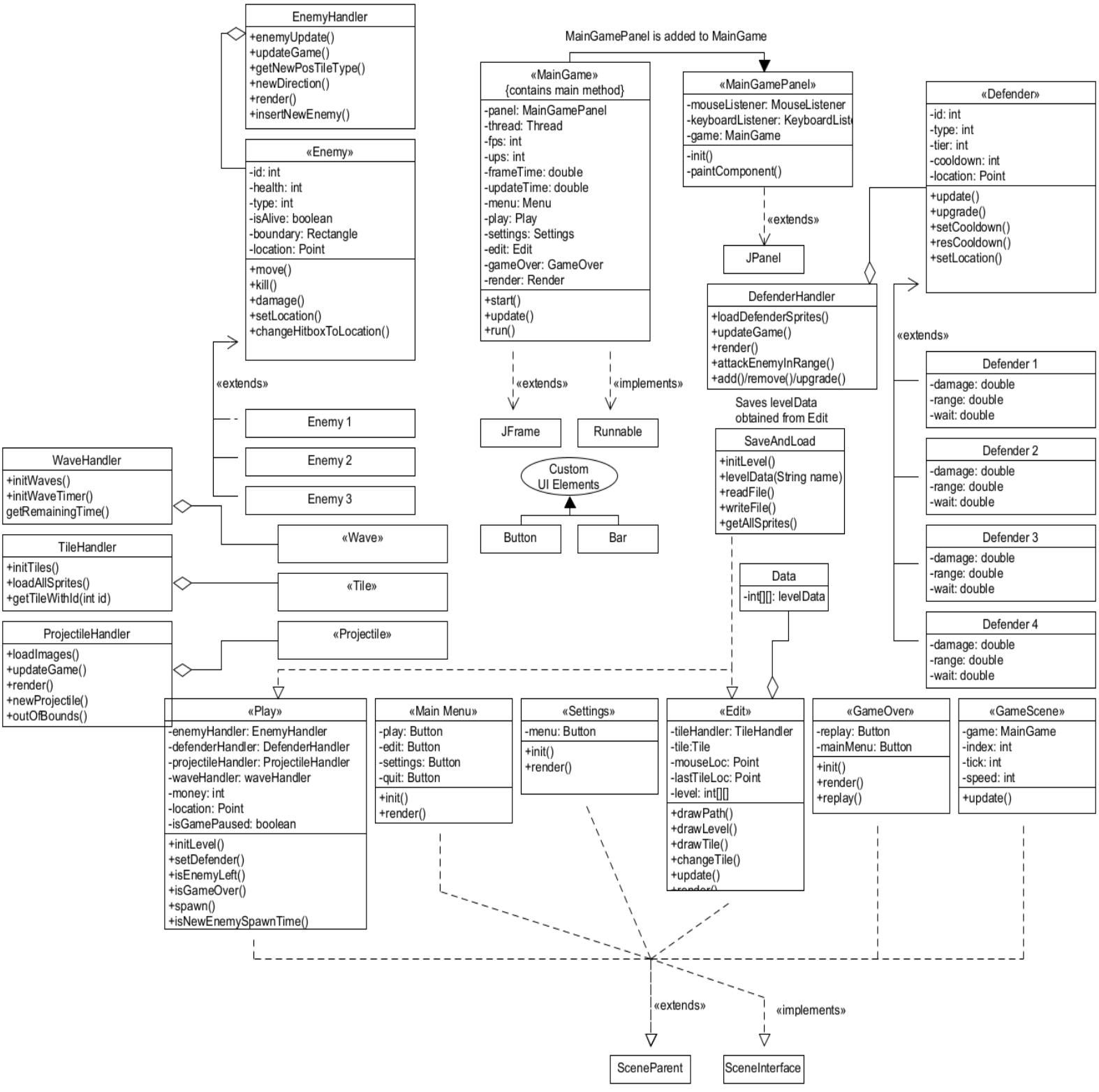
(Version 2.0)

11 May 2022

1. Introduction

Defender of the Galaxy is a space-themed tower defense game in which the player purchases several constructions and special abilities in order to defend the base against stacking enemy waves. In each round, several enemy units will follow a particular path through the map to reach the base. The player must strategically locate his weapons and use the special abilities to defeat the enemy waves.

2. Details



2.1. Enemies :

An abstract class for all enemy types. It imports EnemyManager. It consists of methods for moving the unit, killing the particular unit, dealing damage from defenders, setting the location of the enemy, and changing hitbox. It also contains a superclass constructor for enemy objects and other setters and getters for its variables.

Enemy Type 1

Extends enemy class.

Enemy Type 2

Extends enemy class.

Enemy Type 3

Extends enemy class.

Different enemy types have different attributes in terms of speed, durability etc.

2.2. Main

Main Game

The main class of that game will be running. It extends JFrame and implements Runnable. It has methods for starting the game, updating game status, and running the game.

Main Game Panel

This class extends JPanel and it has necessary elements in it for event handling and creating initializing the game panel.

Keyboard Listener:

This class implements the KeyListener interface. It overrides keyPressed(), keyTyped(), and keyReleased() methods in order to control the game state.

Mouse Listener:

This class implements the MouseListener interface. It overrides the methods from MouseListener interface for controlling mouse events.

2.3. Play

This class is for controlling other entities. It contains all Handler typed classes, the Bar class for the bottom control bar and levelData[][] variable to keep track of the tiles on the screen. It has methods for loading Images, rendering the current scene of the game, updating game state and the rendered scene, updating enemy moves, finding the path for enemy waves, spawning new enemies etc.

Enemy

Functions are controlled by EnemyHandler. It has methods for loading enemy images and all other methods for manipulating enemy units.

Wave

It contains ArrayList of current enemies on the wave. Its functions are controlled by WaveHandler, which has methods for spawning new waves, tracking wave timers, checking the enemy count, resetting etc.

Projectile

It contains int ID, type, damage; Point2D location, double rotation and double xChange, yChange variables. Its functions are controlled by ProjectileHandler,which contains ID variable for the projectiles and ArrayLists of all projectiles and explosions and has methods for controlling the projectiles.

Tile

Functions are controlled by TileHandler. It is for creating the map and the path that enemies will follow.

Defender

Functions are controlled by DefenderHandler. It has methods for loading defender images, adding and removing defenders, upgrading, attacking enemy units etc.

2.4. User Interface

Button

Custom buttons that are needed for gameplay.

Bar

Custom control bar for the gameplay. It includes the market, game information, time for new wave etc.

2.5. Interfaces

2.8.1 Scenes Class: Other classes in this category will extend this class.

2.8.2 Settings Class: This class will have menu Button and two methods:

init() and render()

2.8.3 Play Class: This class has three managers: enemyManager, defenderManager and projectileManager. It has also money as int, location as Point, isGamePaused as boolean.

The methods in this class are as follows:

setLevel(), setDefender(), isEnemyLeft(), isGameOver(), spawn(), isNewEnemySpawnTime()

2.8.4 Game Over Class: This class has replay and mainMenu buttons and three methods:

init(), render() and replay()

2.8.5 Pause Class: This class has methods to get into settings interface and methods to resume the game or exit the game.

2.8.6 Market Class: This class will have get and set methods for the prices and the players money.

2.8.7 Edit Class: This class is about editing and setting the game.

Variables in this class are as follows:

Tile tile;

Point mouseLoc;

Point lastTileLoc;

level int[][];

Methods in this class are as follows:

drawPath(), drawLevel(), drawTile(), changeTile(), update(), render()

2.6. Defender

An abstract class for all defender(defense tower) types. It consists of get and set methods about the damage, range, fire rate, location and the tier of the defender.

It also has a method to upgrade the defender.

Defender Type 1:

Extends defender class and has special features like a slow.

Defender Type 2:

Extends defender class and has special features like a burn effect.

Defender Type 3:

Extends defender class and has other features.

Defender Type 4:

Extends defender class and has other features

3. Task Assignments

Mert Fidan	Play Main Interfaces Handlers
Tolga Han Arslan	Enemy Defenders Handlers
Umut Arda Filiz	Waves Projectiles Handlers
İrfan Hakan Karakoç	UI elements Tiles Handlers

4. Conclusion

This report outlines the design and implementation of our project.