|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **CS102** | | **Fall 2017/2018** | | | Project Group | 15 |
| Instructor: | | **Uğur Güdükbay** | | |
| Assistant: | | **Yalım Doğan** | | |
| **Criteria** | **TA/Grader** | | **Instructor** |
| Presentation |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
| Overall |  | |  |

**~ Tower Defense Game ~**

Snowball

**Melih Ünsal 21501085**

**Erkin Beşer 21400961**

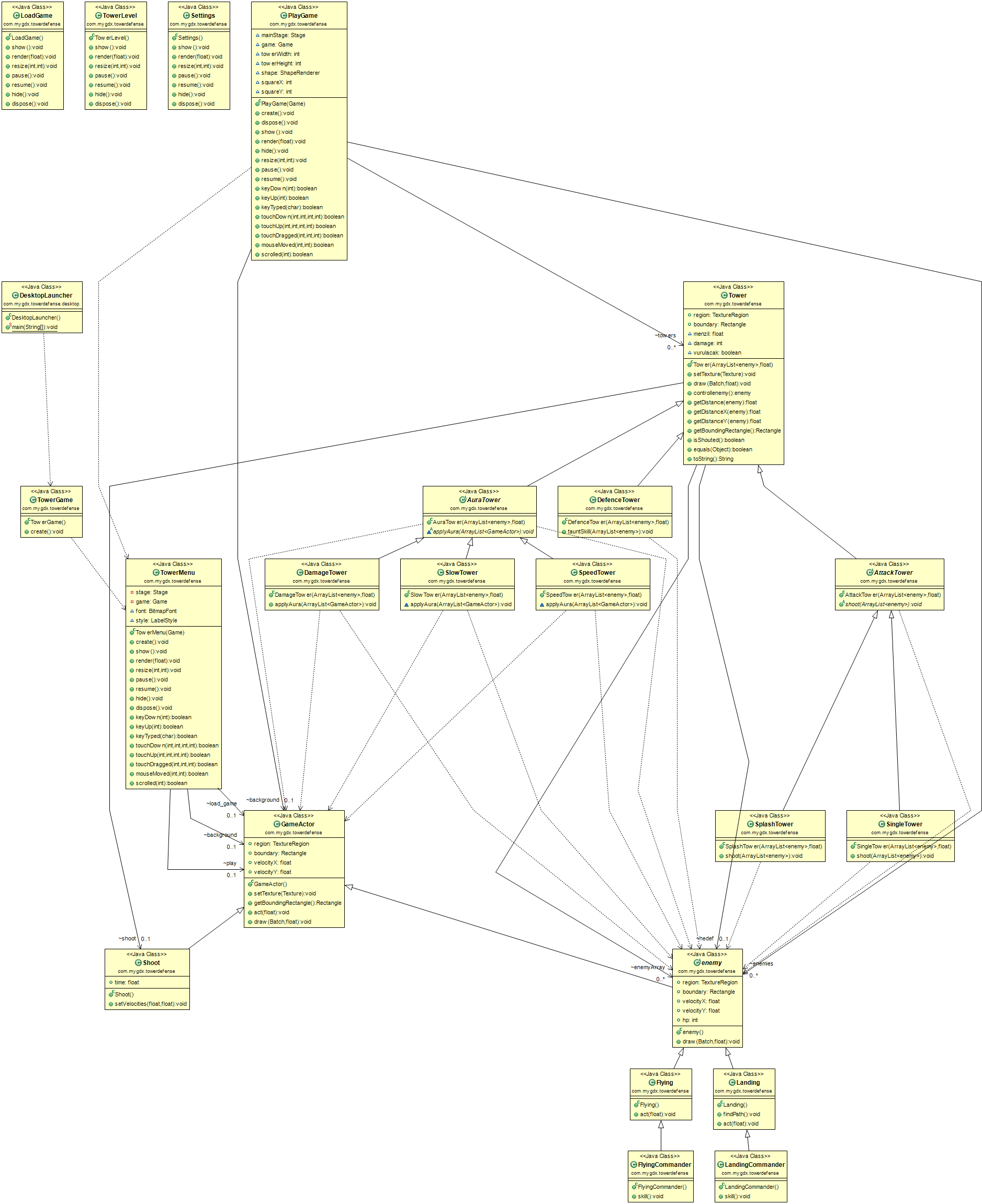
**Burak Alaydın 21603427**

**Can Savcı 21300803**

|  |
| --- |
| Detailed Design Report  **24 November 2017** |

**1.Introduction**

We are going to create a single player tower defense game which will be different from the classical tower defense type. It is going to be a desktop game, written in java. The user will be allowed to build his towers anywhere he wants on the map. In this way, he will create his own maze that requires advanced strategies than the classical types require. The game will be challenging because of its AI that can change its strategy in different situations. We use A\* algorithm so that our units which is under Enemy Class will find their path by using A\* algorithm. And also we use free Assets to visualize our tower and enemy units.

**2.Details**

Our project will be a Desktop Project and we will use database to create Scoreboard so that players can have track their progress in the game. We have three different types of towers which are AuraTower, DefenceTower and AttackTower classes. AuraTower class has three child class which are DamageTower, SlowTower and SpeedTower their applyAuro() methods are different so they are different classes. On the other hand, AttackTower class has two child which are SplashTower and SingleTower their shoot() methods are different so that they are different classes. Also we have enemy units which are under the enemy classes. There are different types of enemy units which are Flying and Landing subclasses of the enemy class and their act() methods are different. For example, for Fyling objects towers are not an obstacle so that Fyling enemy units will fly over all towers and just follow the path in the game (which means for Fyling enemy units, complication of the maze is not important). Flying class has subclass which is FlyingCommander. Landing classes has subclass which is LandingCommander. These subclasses has skill() method so that they are different than the Flying and Landing objects. Their skill() method can affect nearby units of the LandingCommander or FlyingCommander.

**2.1 Using References and Outline**

We have searched so many tower defense game models and maps and at the end we have inspired from the very famous game called Warcraft III: The Frozen Throne. Since it has so many tower defense game maps, we have looked those maps to create our own game map. Absolutely, we didn't do any copy past from those maps, we actually consider the benefits and shortcomings of them, then we create our own maps. We also benefit from LibGDX library of Java which is very useful for game development. We have used some PDFs about this Java library while developing our game. While creating our UML Diagram we used UML Tool which is called “The ObjectAid UML Explorer for Eclipse” from website http://objectaid.com.

**3.Summary & Conclusions**

We have searched so many tower defense games to see how to design a game map and also how to design a tower defense game. After we study these games, we have determined the benefits and also shortcomings of these games, then we have created our own UI. In our group we divide our project work between our group members. However, each group member will work on the GUI which means we create our GUI with group work instead of individual work. (However, each individual of the group can also work on the GUI in his free time.)  
  
Project Task: In our group, all individuals will have good understanding of each part of the project. Each member has its own focusing part in the project, at the same time other group members also have knowledge about other parts of the project. Thus, we divide some parts of the project between our group members it is as in the following:

Melih Ünsal : GUI, implementing Tower classes (also their child classes), finding Assets for visualization of our Towers and Enemies

Burak Alaydın: GUI, implementing Tower classes (also their child classes), implementing Shoot class

Can Savcı : GUI, implementing Enemy classes (also their child classes), work on database for ScoreBoard in the game

Erkin Beşer : GUI, implementing Enemy classes(also their child classes), implementing A\* algorithm (so enemy units in the game will find the shortest path)

[[1]](#endnote-1)**Reference**

1. “The ObjectAid UML Explorer for Eclipse.” ObjectAid UML Explorer, objectaid.com/. [↑](#endnote-ref-1)