Documentaion of

Tower Game

*Shahriar Rostami*

*Sasan Farid*

# Game Description

In this project, we are building a real time Tower Defence game. The goal of this game is that when a number of critters enter the map, trying to reach the other side, the user tries to kill them before they reach the other side by putting towers across the map which shoot those critters. And by real time we mean that the user can add, remove or upgrade any tower while the game is being played.

The user has the option to play on a randomly generated map by simply clicking on “Start” button on the screen or to load a previously save map on the hard drive using the “Load Map” button and then start the game by clicking “Start”.

The user is given an amount of $1000 to start with which can be used to build towers and upgrade them. The towers can be put on the map using the mouse click on which the user can select between three different type of towers (Regular, Mass and Riffle), and pressing “Alt+Click” on each of them shows the tower’s menu which include removal, upgrading, the strategy for that tower, and showing the values of the type of the tower, money spent on it, etc.

A regular tower, shoots the critter at slow time intervals, a mass type tower has shorter time intervals and reduces more healthpoint from the critter and riffle has those values in between.

For each tower at any point, by openning the menu window for that tower, we can select different strategies to shoot at critters:

* “Closest”: shoots the critter which is closest to the tower, and if two critters are at the same distance, takes turns.
* “Farthest”: chooses the farthest critter from the tower and shoots it.
* “Sickest”: chooses the critter with less health points among those within range of the tower.
* “Healthiest”: shoots at the critter with the highest health points among those within the tower’s range.

Each tower can be upgraded to as much as level 5. In each upgrade, an amount of money is payed, and the range or the shooting amount is raised.

The critters start moving from one side of the map, and try to go through a path to get the other side. Each critter has a health value which is at its maximum value and is decreased while being shot by a tower. And if it becaoms zero it cannot move any furthur in the map.

The user has won a level in the game if he/she kills all the critters in that level before they reach the other side of the map. And loses the game if even one of the critters reach its destination.

# Programming Enviroment

In this project we did our coding on JavaSE-1.7. We used JavaDoc for documentaion, Java Swing for GUI, Github was used as the repository, and Junit(4) for testing the codes.

# Details on codes

This program uses two different *thread*s to run. One thread is used for the windows of the game, while the other is

The program starts by running main(String[] args) in class Runner. This is the main function of the whole project. In this class we build our windows, buttons, their functionalities and also the engine of the game.

The buttons shown on the screen are:

* “Start/Pause”: this button starts the game and can also pause it.
* “New Map”: this button creates a random path.
* “Create Custom Map”: this button shows the panel to the user in order to make his/her own customed map.
* “Load Map”: this button loads a saved map on hard disk and the player can play on that map.
* “Save Map”: saves the current map on hard disk.

The Engine class is the brain for this game. It manages the acounting of the player, creates the critters and runs the surface of the game. Also it creates the first map needed for initialization of the game.

The *acounting* is so that the player starts the game with $1000. By buying his first tower (the value of a level 1 tower is $100), his/her money becomes 900 and so on. Selling each tower returns half of its value to the player. The user can buy towers until his/her money becomes zero.

The *critters* have an initial health value 100 and each time they are hit by a tower’s shooting, this value gets to the 90% of the previous health value; i.e. the first shooting against a critter sets its health value to 90.

The Surface class is responsible for Graphics of the gameplay.

The map for game is being built in MapGenerator class. This class also produces the path from the starting point to the end point. This class has different methods for a randomly generated map, a user-driven generated map and saving and loading into/from files. The Map class is the correspondent for interaction of MapGenerator class and the GUI for showing of the map and also validation of it.

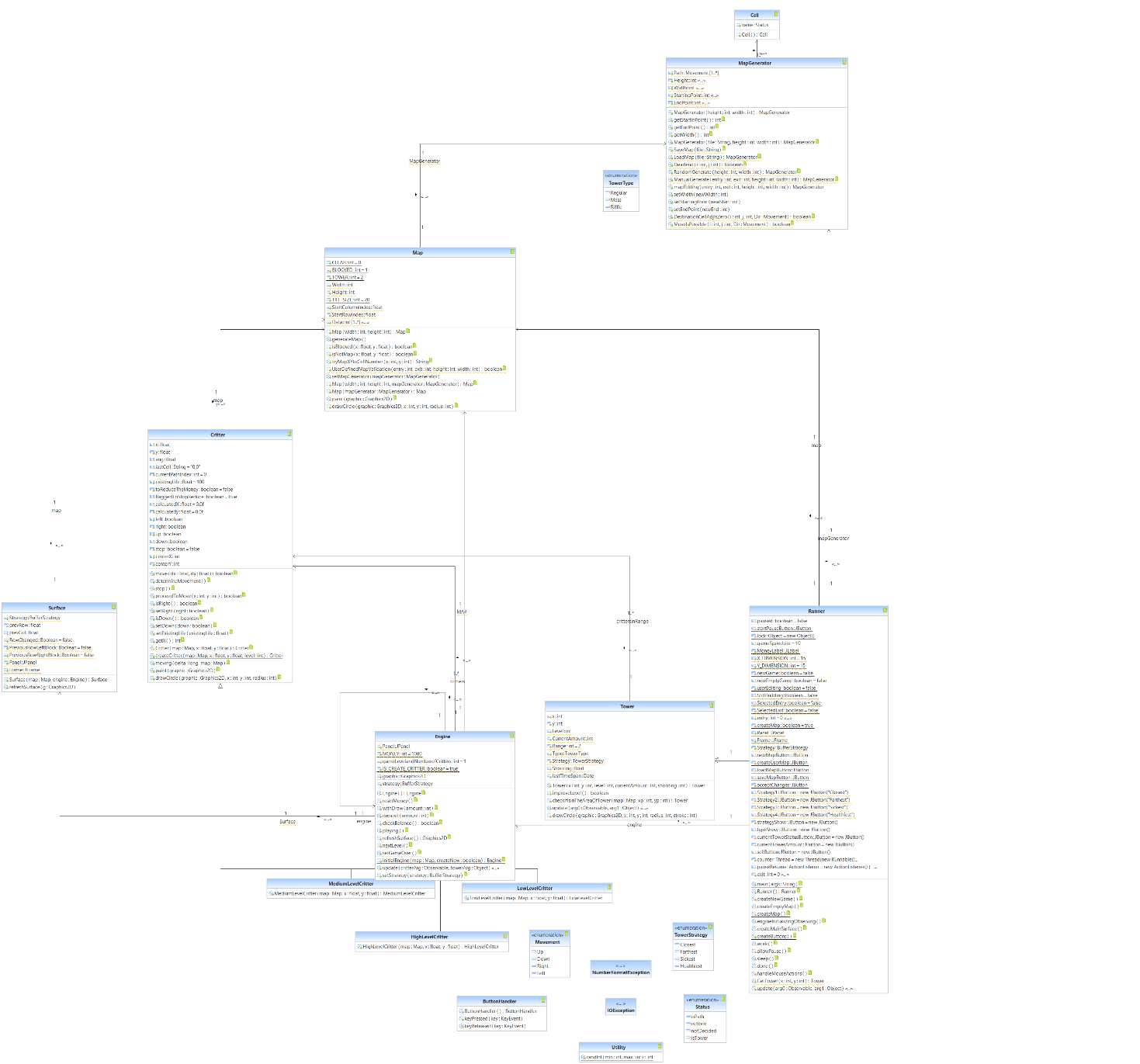
# Design Patterns

The runner is observing the changes in the engine which checks the values for accounting and also the movement of critters. Also in runner we keep a list of critters for each tower in the map, in order to get the needed critter to be shot at based on the tower’s strategy.

The engine is observer to the critters (movements) and notifies the runner for their location, which should be shown on the map.

In creation of each critter we are using factory pattern. That is when creating a new critter, we look to the game level, and based on that, create a low level critter, medium level, or high level critter. And also based on the same value, the number of critters to enter the game change.

# UML

This UML shows the classes and their interactions in the program.

This UML was created using Rational Rose.