Object Oriented Software Design Project – Monopoly

Requirement Analysis Document

Authors: Mertcan Karaca, Mehmet Mum, Erhan Güngör

Version: 0.1

Date: 14.10.2017

TABLE OF CONTENTS

1. Summary of the Project
2. Project Description
3. Requirements for First Iteration
4. PROJECT SUMMARY

Monopoly is a well-known board game where players roll dices to move around gameboard, buying and trading properties and develop them with houses and hotels. The main purpose of the game is being drive the other players to bankruptcy. A player collects rent when another player comes to that player’s property. Money can also be gained or lost from Chance and Community Cards and tax squares. The player may end up in jail, which they can not move until some of different conditions happen. Our main goal is making a working simulation of this game

1. PROJECT DESCRIPTION

The main goal of our project is creating a Monopoly game simulator so that the user can observe the game and learn how to play it. We are using Java in this project and it is going to be a script-based project.

1. REQUIREMENTS FOR FIRST ITERATION

Firstly, we need to start with creating the real world game objects such as players,pieces,dice and board. So, we should describe our Player object firstly. There are multiple players and to avoid conflict we need to define an attribute named playerID to keep the data who’s turn it is. With a player id, we need to give the player a piece to play it on the field. So that we should describe Piece object after that. On the board, the piece is in a square, so it has a location. Based on this, we need to create currentLocation attribute for our Piece. On the board, there are 40 squares and each one of them have different locations and attributes. So, we should define our Board object and under that Square object connected to Board. All different squares have a common attribute and that is they have a location. So, we need to define a squareLocation attribute under Square object. The last thing we need to create to play the game in the simplest form is dice. When you toss a die in real life, there is one face that is always looking up and that is the result of tossing. That’s why we should create a field named faceValue for the result of tossing the die. After defining our objects, we need to create what methods and functions we are going to use. To access the data on each object, we need to create getter and settter methods. To play the game, the current player rolls the dice. So, we needed to create a method named roll that sets face value of die between 1-6. Besides that, when a player rolls two dice and if they are equal, the player rolls dice again. To control this, we should create a method named checkDouble to see whether if dice are equal or not.