# Monopoly Game
# Requirement Analysis Document

## Requirement Specification Vision

We would like to redevelop the well-known Monopoly Game. The game shall be developed as console based one and will be a simulation. The user just starts the game by entering the number of the simulated players and then the game runs itself. The game shall run until one of the players' money vanished.

## Problem Statement

Our team objective is to rebuild the old, familiar Monopoly Game from the rough, using Java.

## Scope

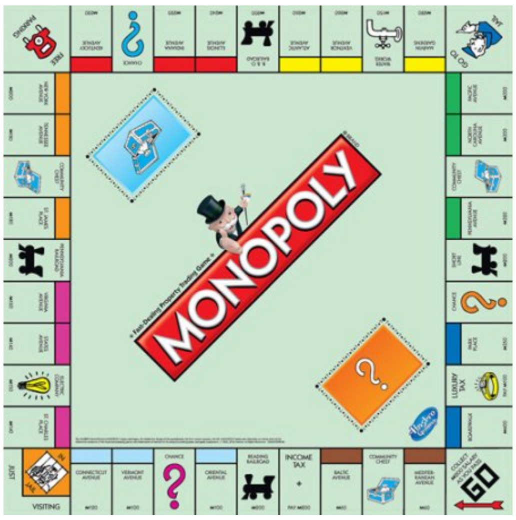

Figure 1: Sample Monopoly Layout

The Monopoly is a game in which players take turns and with the help of two equal dices make moves around a square-shaped game board as shown in Fig.1. The game board consists of smaller rectangles that have specified functions or meanings for the game mechanism. Game can be played minimum two players and up to eight players maximum. Each player starts his/her turn by rolling the dices and then move on as total numbers of the dices. The game runs itself; there will not be any real player, which plays the game. After each iteration, the actual state of game and players will be displayed from the console.

## System Constraints

Will run as a console application, there is no need to additional software to be installed except the Java IDE.

Allow up to 8 players to play maximum, 2 players minimum.

Will provide the current state after each iteration, the finishing scores, winners and losers of the game.

## Stakeholders

Murat Can Ganiz (Customer)

Mehmet Ali Sakallı

Muhammed Kılıç

Onur Bayraktar

## Rules

- Everyone starts with $1500.
- If total assets of a player < $0, then he/she loses.
- Minimum 2 players, maximum 8 players.
- There will be two dices. With the total value of dices, next player moves on clockwise around the board the corresponding number of squares.
- If the player completes his run around the game board and passes the starting point, will collect 200$ from the bank.
- If one player rolls doubles, he rolls again. If one player rolls doubles three times in series, he will be waiting at the jail square to other players move 3 times. However, if the punished player rolls a doubles at this time, he can leave the jail square otherwise, he must pay $50 after his punishment is over.
- There are some different kinds of squares in the game ;
  1) Properties
  2) Tax/Income Squares
  3) Jail Squares
  4) Card Squares
  5) Starting Point
- If a player lands on an unclaimed property, he can buy the place by paying the buying price.
- Players can build houses and hotels on these properties and from these properties and the lands; they will collect some money from the other players.

## Glossary of Terms (Alphabetically Listed)

Bank: The structure that keeps the money and disturb it to the players.

Card: The set of structures that consist of different instructions about the game.

Dices: Two equal sized and shaped plastic cubes, which the game can be played by shaking and throwing from one to six.

Game board: The board that contains squares players' pawns and the dices.

Money: The numerical values that each player has, and the foremost feature of the game to decide who won/lose.

Piece: A symbol that belongs to the player, which can be a car, horse, hat, ship etc. represents him in the game.

Player: A simulated person who plays the game and take actions.

Starting Point: The Square that each player locates at the starting of the game, players collect moneys when they passed over it.

Status of Game: The information part for the user about what is the latest scores, locations, properties etc. of each player.

Square: The places that properties, cards and starting point are located, players can move on them.

## Use Cases

| | |
|---|---|
| Name | StartNewGame |
| Actor | User, System |
| Description | User starts the program |
| Condition | 1.Application is running<br>2.The game is not currently in progress |
| Flow of events | 1. The user stars the program<br>2. System asks the user: "How many players will be playing the game?"<br>3. User enters the number of players. |

| | |
|---|---|
| Name | TakeTurn |
| Actor | System |
| Condition | Application is running<br>The game is in progress |
| Flow of events | 1. The system decides in which player should move on the next time<br>1a. If it is the starting of the game, then it is Player1's turn.<br>1b. If it is not the start, then it goes to the next player. |

| | |
|---|---|
| Name | RollDice |
| Actor | System |
| Description | The system rolls the dice. |
| Condition | Application is running<br>The game is in progress |
| Flow of events | 1. After user started the program or all the players' turns are completed, dices are rolled<br>2. System computes the total value that seen on dices.<br>2a. If the player rolls a double, continue with the same player.<br>2b. If the players rolls doubles three times in series, will go to jail square.<br>2c. If none of them above, continue. |

| | |
|---|---|
| Name | MovePlayer |
| Actor | Player |
| Condition | Application is running<br>The game is in progress |
| Flow of events | 1. The dices were rolled; it is time to move the player to corresponding square.<br>2. The players' pieces will be moved on to the new square location such as : CurrentLocation + TotalDices<br>2a. Check for "RollDice" case (2a and 2b) if the player will go to jail or not. |

| | |
|---|---|
| Name | ShowCurrentStatus |
| Actor | System |
| Description | The system shows the current status of the players and the game to the user. |
| Flow of events | 1. The system prints the following information to the screen after each round of the game:<br>Player(Number) / Piece / Location(Square) / Money($) |

| | |
|---|---|
| Name | GameEnd |
| Actor | System, User |
| Description | One of the players are out of money, or the round limit has reached! |
| Condition | Application is running.<br>The game is not in progress. |
| Flow of events | 1. The system printf the result of the game to the screen.<br>2. The user selects the Exit option<br>3. The system asks for confirmation with confirmation form. |