Sean Mitchell, Megan Haskins, and Ansley Solomon

UAH Monopoly

CS 321-02 Spring 2019

Contents

[Introduction 2](#_Toc6819488)

[Team Composition 2](#_Toc6819489)

[Design Goals 2](#_Toc6819490)

[Game Design 3](#_Toc6819491)

[Design Choices 3](#_Toc6819492)

[Design Patterns Used 3](#_Toc6819493)

[UML Diagrams 3](#_Toc6819494)

[Uses Cases 3](#_Toc6819495)

[CRC Cards 4](#_Toc6819496)

[Operation 5](#_Toc6819497)

[Starting the Game 5](#_Toc6819498)

[Gameplay Instructions 5](#_Toc6819499)

[Victory Conditions 5](#_Toc6819500)

[Defeat Conditions 5](#_Toc6819501)

# Introduction

For our final project, we recreated the board game Monopoly using Java. The reason for making this game was to recreate the experience of playing monopoly on a computer.

## Team Composition

Our team composition and labor division are presented in the following table.

|  |  |
| --- | --- |
| Name | Responsibilities |
| Sean Mitchell | Backend, Integration, Javadoc |
| Megan Haskins | GUI, Integration, Testing |
| Ansley Solomon | XML, Javadoc, Documentation |

## Design Goals

|  |  |
| --- | --- |
| Primary Objectives | Secondary Objectives |
| GUI |  |
| All data read from XML |  |
| Dynamic Card and tile Events |  |
| Multiple players |  |

# Game Design

## Design Choices

Several key design choices were made in the course of prototyping this project. The choice to have only local multiplayer instead of networked multiplayer was made to more accurately simulate the experience of many people around a game board.

## Design Patterns Used

Two design patterns were used in design this project. Several other patterns were considered, such as Singleton and Interface, but the choice to go with the following patterns was made due to how the classes were going to be used.

The first was a Passive Model View Controller or Passive MVC. This approach was used to design the interfaces and relationships between the GUI the player sees and the backend that performs computations based on received player input.

The second was an Object Adapter. This pattern was applied to the Tile\_Adapter class. This class served as an interface that allowed the game board and player classes to interact with the seven different tile classes through a common interface.

## UML Diagrams

# Uses Cases

# CRC Cards

# Operation

## Starting the Game

To start playing the game, simply load the project and select the “Start game” button. You will be asked to enter the number of turns to play and the names of the four players. There are no accounts required, so you may just enter any String to use as player’s name.

## Gameplay Instructions

The gameplay is the same as a standard game of Monopoly. Each turn, that player will roll a dice and move to the tile that is the resulting roll away. If this tile has a special event, such as a drawing a card, the player will receive the result of this event.

If the tile is a property tile, the player may buy the tile. If this property tile is already owned, the player will pay rent to the current owner of the tile.

## Victory Conditions

The game ends when the turn counter set at the start of the game is reached or all but one player have been eliminated. At this point, the remaining player with the highest score is declared the winner.

## Defeat Conditions

A player is defeated when their money reaches a negative value. They are informed they are eliminated and are unable to continue playing rounds.