Megan Nicole Haskins

CS 321-02 Final Project – UAH Monopoly

CS 321-02 Spring 2019

Contents

[Introduction 2](#_Toc6847998)

[Team Composition 2](#_Toc6847999)

[Workflow 2](#_Toc6848000)

[Design Goals 2](#_Toc6848001)

[Operation 3](#_Toc6848002)

[Starting the Game 3](#_Toc6848003)

[Gameplay Instructions 3](#_Toc6848004)

[Victory & Defeat Conditions 3](#_Toc6848005)

[Game Design 4](#_Toc6848006)

[Design Choices 4](#_Toc6848007)

[Frontend 4](#_Toc6848008)

[Backend 4](#_Toc6848009)

[Information Storage 4](#_Toc6848010)

[Design Patterns Used 4](#_Toc6848011)

[UML Diagrams 5](#_Toc6848012)

[Class diagrams 5](#_Toc6848013)

[Sequence Diagrams 6](#_Toc6848016)

[Sequence Diagram 1: Player rolling a dice on their turn 6](#_Toc6848017)

[Sequence Diagram 2: Player trading property to another player 6](#_Toc6848018)

[Use Cases 7](#_Toc6848019)

[Meeting Minutes 8](#_Toc6848020)

[02/09/2019 Meeting Minutes 8](#_Toc6848021)

[02/15/2019 Meeting Minutes 9](#_Toc6848022)

[03/22/2019 Meeting Minutes 10](#_Toc6848023)

[Appendix 11](#_Toc6848024)

[Software Used 11](#_Toc6848025)

[GitHub link 11](#_Toc6848026)

# 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. Using design patterns and division of labor, this complex problem was broken into three distinct subproblems. These subproblems were the frontend interface, the backend, and the information storage.

## Team Composition

Our team composition and division of labor are presented in the following table. Below are the main responsibilities of each team member, but each member would also assist in other tasks as required.

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

## Workflow

Weekly standup meetings were held to discuss progress, exchange ideas, and perform work. These meetings were consistently held through the term, with around 14 meetings in total. Outside of meetings, work was done remotely and shared through GitHub. The link to the project’s GitHub is included in the Appendix. Communication outside of the meetings was done using the GroupMe application.

## Design Goals

Below are the primary design goals we hoped to meet with during development of this project.

|  |  |  |
| --- | --- | --- |
| Design Goal | Not met | Met |
| GUI |  | X |
| All data read from XML |  | X |
| Dynamic Card and tile Events |  | X |
| Players can interact with game board and each other |  | X |
| Local Multiplayer |  | X |

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

# 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.

## Frontend

A simple graphical user interface, or GUI, was designed and built for this program. Through this GUI, a player would be able to interact with all the required gameplay functions. This GUI also shows relevant gameplay information.

## Backend

The backend functions as an interface between the frontend GUI and XML parsers. Once the XML files were parsed, the backend would create a common interface for the various objects created. The backend was also responsible for computing gameplay, such as a player’s money or location playing.

## Information Storage

To store the information, XML files were used. The final program used two XML files. One file stored information on the Properties and the other stored information on the Cards. These would be parsed by the program, which would create objects and structures for this information.

## Design Patterns Used

Three design patterns were used to design this project. Several other patterns were considered, such as 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.

The third was the Singleton Pattern. The Board class utilized this design so that the program would always reference the same game board. A Board is mainly used throughout class view1 and was pertinent that multiple boards not get instantiated.

# UML Diagrams

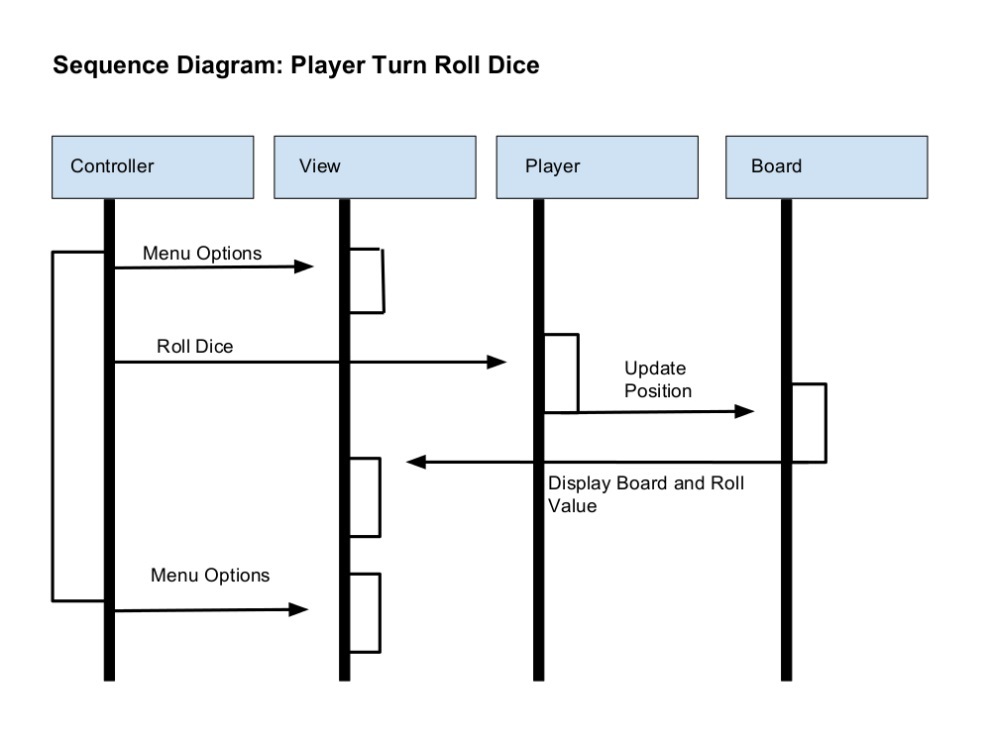
## Class diagrams

# E:\School\2019 Spring\CS 321\Final Project\CS321_SP2019\uml\main_uml_simple.png

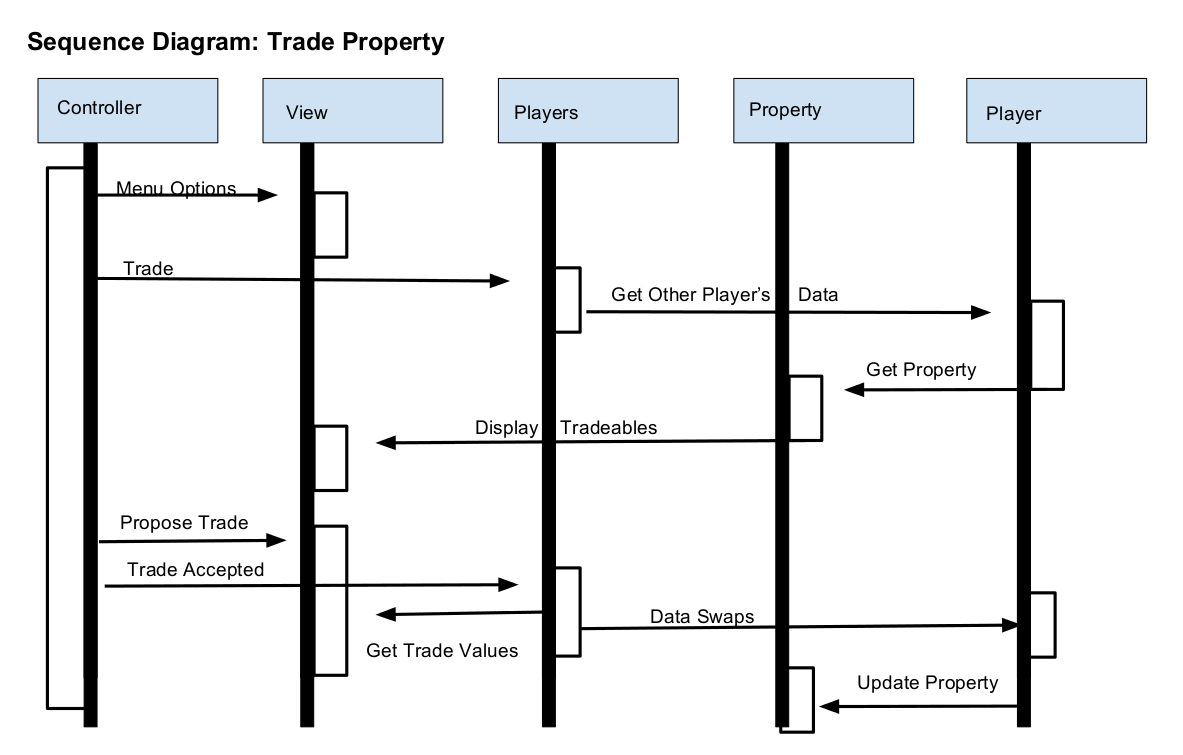
# 

## Sequence Diagrams

### Sequence Diagram 1: Player rolling a dice on their turn



### Sequence Diagram 2: Player trading property to another player



# Use Cases

* Use Case 1
  + Player starts turn
  + Player rolls dice
  + Player lands on a property tile
  + Player buys property tile
  + Player turn ends
* Use Case 2
  + Player starts turn
  + Player rolls dice
  + Player lands on a property tile
  + Property is owned by someone else
  + Player pays rent to that player
  + Player turn ends
* Use Case 3
  + Player starts turn
  + Player rolls dice
  + Player lands on a property tile
  + Property is owned by someone else
  + Player pays rent to that player
  + Player money is now negative
  + Player is eliminated from game
* Use Case 4
  + Player turn ends
  + Turn counter reaches maximum
  + Game ends
  + Player scores are computed and presented
  + Player with the highest score is declared the winner
* Use Case 5
  + Player starts turn
  + Player rolls dice
  + Player lands on a chance card tile
  + Event on card is triggered on player
  + Player is affected by event
  + Player turn ends

# Meeting Minutes

## 02/09/2019 Meeting Minutes

* Moderator: Sean Mitchell
* Time of meeting: 2 PM
* Discussion of goals
  + Map
  + Buy/Trade/Sell
  + GUI
  + Multiplayer aspects
  + Community chest and chance cards
  + Dice rolling
  + Attributes of tiles
  + Networking?
  + Player set time limit
  + Tile graphics
  + Random tile locations?
  + Tokens
    - Horse
    - Rocket
    - Laptop
    - Fedora
    - Glasses
    - Hoverboard
    - Beaker
  + Use Cases
    - Winning
    - Losing
    - What happens if a player drops out?
    - Surrender
    - Networking
* Meeting ended at 3PM
* Next meeting at 1PM next week

## 02/15/2019 Meeting Minutes

* Moderator: Ansley Solomon
* Time of meeting: 1PM
* What has been added to GitHub since last week?
  + Skeleton code
  + Token images and board tile images have been added – need to make background transparent
  + Networking – someone runs it, it makes a host, it has a client handler class client connects to handler, add threads for each user
* To Do:
  + Cushion classes
  + Turn timers
  + XML
  + Card Class
  + Board Class
  + World Class
  + Player Class
  + Driver Class
  + Menu/options
  + Display board
* Work Assignments
  + Ansley – Community chest and chance cards – Design images, list events, etc.
  + Megan – Tile class, board class
  + Sean – XML, networking stuff
* End of meeting: 1:30PM
* Next Meeting: 1PM on Friday next week

## 03/22/2019 Meeting Minutes

* Moderator: Megan Haskins
* Start time: 11 AM
* Topics Discussed
  + Removing network code
  + GUI
  + XML cards
  + What to place in powerpoint
  + CRC cards
  + Utilities and Crosswalks
  + looking over dice roller
  + Deciding to let people know if they decided to do more them their assigned work over spring break
* Division of Labor:
  + Ansley - more XML
  + Megan - GUI and diagrams
  + Sean - back end documentation
* End time: 11:30 AM
* No meeting next week (spring break)

# Appendix

## Software Used

* Netbeans 8.2
* StarUML
* Git
* GroupMe

## GitHub link

* https://github.com/SeanMitchell1994/CS321\_SP2019

|  |
| --- |
|  |