Programming Technology 1

Adnan Adnan (NBDK4L)

Capitaly Board Game – Assignment (3)

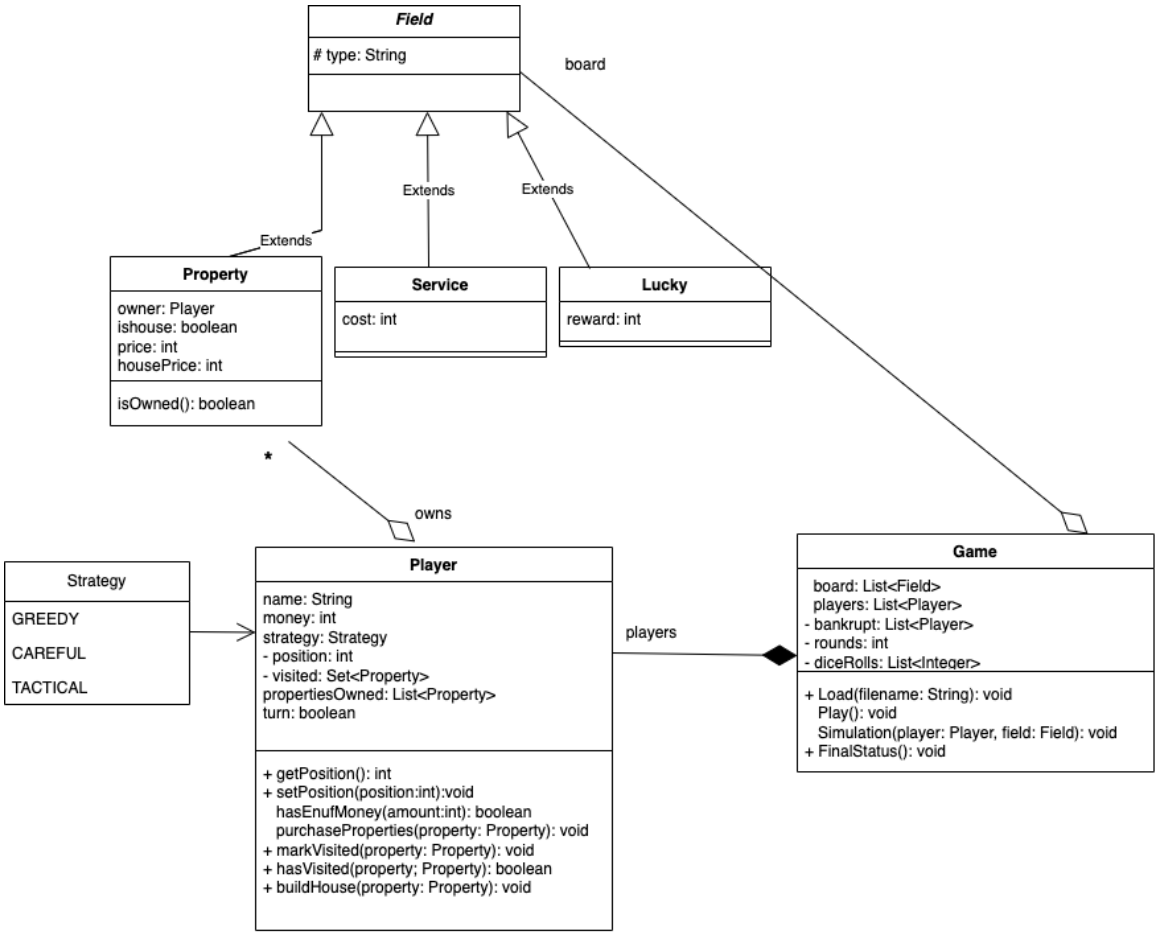
Description of the Task:

In this assignment, the goal was to simulate a Monopoly-like board game where players move around the board, purchase properties, pay rent, interact with services, and try to avoid bankruptcy. Each player starts with an initial balance of 10,000, and the game proceeds for a series of rounds. Players roll dice and move a number of spaces on the board, landing on different types of fields such as properties, services, and lucky spots.

Players may buy properties, pay for services, or gain/lose money through lucky spots. The game ends when a player goes bankrupt or after a predefined number of rounds.

Class Diagram:

The following diagram shows the relationships between the main classes of the game. This UML diagram illustrates how the classes are organized, including the Game class, Player class, and different types of fields like Property, Service, and Lucky.



Method Descriptions:

- **Game class**

Play Method:

* This method controls the flow of the game. It iterates through a predefined number of rounds, allowing each player to roll
* the dice and move accordingly on the board. Based on the type of field the player lands on (property, service, lucky),
* different actions are triggered, such as buying properties, paying for services, or gaining money through lucky fields.

Load Mehtod:

* Loads the game data from a specified file. The file should contain information about the fields, players, rounds, and dice rolls.

* Throws IOException if there is an issue reading the file.

Simulation:

* Simulates the actions that a player takes based on the field they landed on. Actions vary depending on whether the field is a property, service, or lucky field.

♦ Player Class

HasEnufMoney Method:

Checks the player has enough money to afford a specific cost. This method is used before performing actions that involve spending money, such as buying a property or building a house.

PurchaseProperties:

This method handles the process of purchasing a property based on the player's strategy. It performs the following actions:

1. If the property is already owned by someone else, the method returns.
2. If the player is "Greedy" and has enough money, they will buy the property.
3. If the player is "Careful" and has enough money, they will buy the property, but only if the property price is less than or equal to half their available money.
4. If the player is "Tactical" and has enough money, they will buy the property, but only on every other opportunity.
5. For "Tactical" players, it toggles their turn flag to skip the next buying opportunity.

hasVisited:

Checks the player has visited a specific property before. implementing the rule where players can only build a house on a property they own after visiting it again.

markVisited:

Marks a property as visited by the player. This is used to keep track of properties that the player has stepped on.

buildHouse:

Handles the construction of a house on a property that the player owns. The player can only build a house if:
They own the property.
There is no house built on the property.
The player has enough money to afford the house construction cost.

Testing:

- **White box testing**

Input Files:

1. Input File 1:
5
Property
Service 2000
Lucky 500
Property
Stalk 3000
3
Adnan Greedy
Hammad Careful
Ali Tactical
3
1 5 2
1 2 3
2 3 4

This input tests invalid property type

```
[Running] cd "/Users/adnansamore/Documents/JAVA/Monopoly/" && javac Main.java && java Main
Exception in thread "main" java.lang.IllegalArgumentException: Unknown field type Stalk
    at Game.Load(Game.java:40)
    at Main.main(Main.java:7)
```

2. Input File 2:
5
Property
Service 2000
Lucky 500
Property
Service 3000
3
Adnan Greedy
Hammad Careful
Ali Tactical
3
1 5 2
1 -2 3
2 3 –4

This input tests negative dice rolls

```
[Running] cd "/Users/adnansamore/Documents/JAVA/Monopoly/" && javac Main.java && java Main
Exception in thread "main" java.lang.IllegalArgumentException: Dice roll must be > 0
    at Game.Load(Game.java:77)
    at Main.main(Main.java:7)
```

3. Input File 3:
5
Property
Service 2000
Lucky 500
Property
Service 3000
3
Adnan Greedy
Hammad Careful
Ali Tactical
3
1 1 3

5 3 3
4 3 4

This input tests game mechanics and simulation

```
[Running] cd "/Users/adnansamore/Documents/JAVA/Monopoly/" && javac Main.java && java Main
Round === 1
Adnan rolls 1 and moves from -1 to position 0
Adnan bought a property for 1000
Hammad rolls 1 and moves from -1 to position 0
Hammad paid 500 to Adnan
Ali rolls 3 and moves from -1 to position 2
Ali received 500 from lucky field.
Round === 2
Adnan rolls 5 and moves from 0 to position 0
Adnan built a house.
Hammad rolls 3 and moves from 0 to position 3
Hammad bought a property for 1000
Ali rolls 3 and moves from 2 to position 0
Ali paid 2000 to Adnan
Round === 3
Adnan rolls 4 and moves from 0 to position 4
Adnan paid 3000 to the bank.
Hammad rolls 3 and moves from 3 to position 1
Hammad paid 2000 to the bank.
Ali rolls 4 and moves from 0 to position 4
Ali paid 3000 to the bank.
-----FINAL STATUS-----
Adnan - Money: 4500
Properties owned: 1
Hammad - Money: 6500
Properties owned: 1
Ali - Money: 5500
Properties owned: 0
```