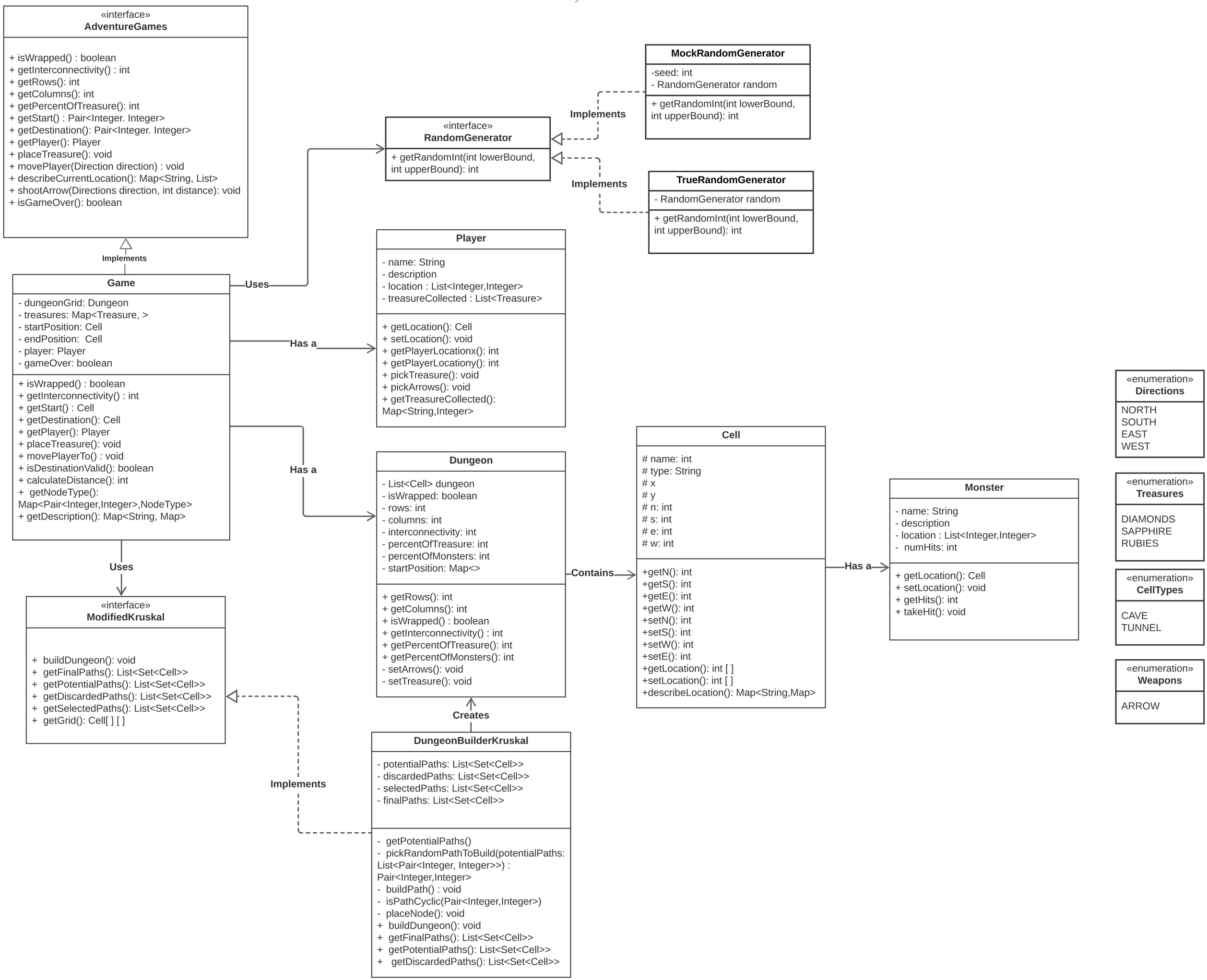
Project 4: Text-based Adventure Game UML Diagram

Lakshyana



Project 4: Text-based Adventure Game Testing Plan

Objective:

This project is the second part of the process of implementing a controller for an adventure game, that takes in the model for a Dungeon game implemented in the previous project, with a few updates to the functionalities. This project be implementing a simple text-based controller that can be used to play the game. In addition, we will be adding enhancements to the dungeon to make the game more interactive and fun.

Testing Strategy

- Test each component of the model, and test the model with the new functionalities added.
- Invalid parameters check and testing will null objects
- Testing the methods using random instance for true random results
- Testing the methods using random instance by passing seed value for testing other functionalities.
- Implement separate classes to test each class methods (Game, Dungeon, Player, Monster, Dungeon Builder, Random Generator classes).
- Implement a class and helper methods to facilitate testing with multiple objects.

Test Cases

1. **DungeonCreationTest()**:
    i. Wrapping and non-wrapping type test
    ii. Varying degrees of interconnectivity
    iii. Different sizes of dungeon grid
    iv. Test if exception thrown for invalid inputs (eg. if grid smaller than the required minimum distance between start and end nodes)
2. **PlayerAndMonsterTest()**:
    i. Test moving player to different cells
    ii. Test getting possible move options
    iii. Test collection of treasure and arrows.
    iv. Test shooting a monster
    v. Test entering the location of a wounded and not wounded monster to test attack and escape from a monster.
3. **RandomGeneratorTest()**
    i. Test if true random generator gives random values
    ii. Test for invalid and null instances
    iii. Test for mock random generator using seed to see if does give the same sequence of numbers for a particular seed value.
4. **Game Test**:
    i. Test a full iteration of a game starting from creating a wrapping dungeon, to placing a percentage of treasures to random caves, placing of arrows to random tunnels and caves, and shooting a monster with arrows.
    ii. Test another similar iteration but for a wrapping dungeon with a degree of interconnectivity greater than 0.
    iii. Test that a player can successfully reach the destination cave when after attacking the Monsters.
5. **ConsoleController Test**:
    i. Complete two iterations of a Dungeon game similar to the Game class test, using a controller.

Operations

The test cases will aim to test and validate the required operations for the game to work run correctly in all scenarios.

Dungeons

The following are the requirements for dungeons:

- Consists of tunnels (with 2 entrances) and caves (1,3, or 4 entrances) that can be traveled from cave to cave.
- Each cave to be connected to at most 4 other locations: North, South, East, West.
- Should be able to be represented on a 2-D grid.
- Have a path from every cave in the dungeon to every other cave.
- To be constructed with a degree of interconnectivity representing the number of paths between caves (0 meaning only one path from a cave to every other cave)
- Locations may or may not "wrap" from one side to the other.
- They can be both wrapping and non-wrapping type with varying degrees of interconnectivity.
- Two caves to be randomly selected as the *start* and the *end* respectively.
- Minimum length of path between the start and the end is 5.

Dungeon: Cave

- Two caves are selected at random as the start and end locations.
- Cave may have one or more treasures.
- Cave may have arrows
- The starting cave should not have a Monster, and the end Cave should always have a monster, and more monsters may be found in other random caves.

Dungeon: Tunnels

- A variation of cave, in that it have 2 entrances, and doesn't hold treasure, or a monster, but may hold arrows.

Treasures

- Types: Diamonds, Rubies, and Sapphires
- Random treasure to be added to at least a percentage of caves specified by the user.

Player

- Place a player at a random starting cave location
- Provide a description of a player's location, including at the minimum a description of treasure in the room, and the possible moves (N, S, E, W) that can be made from player's current location
- Allow a player to move from the current location
- Allow a player to pick up treasure found in their current location.
- Allow a player to pick up arrows found in their current location.
- Allow a player to smell the monster near their current location.

Monster

- Place a monster at the destination cave location, and additional monsters at random Cave locations.
- Allow a monster to take a hit when an arrow is shot by a player, if it lands in the Monster's location.

- Allow a monster to attack a player that enters its current location, unless it's wounded from a prior attack, in which case, the player should have a 50% chance of survival.