2017-04-26

Interaction Diagrams

Assignment in the course PA1435, Objektoriented design

Igor Radulovic, Fredrik Engvall, David Fröström, Sara Huslid, Jonathan Andersson

Name	Social Security	Contribution Thinking (in %)	Contribution Writing(in %)
Igor Radulovic	940524-8858	20	20
Fredrik Engvall	930329-2453	20	20
David Fröström	930222-5512	20	20
Sara Huslid	950804-1002	20	20
Jonathan Andersson	900522-2556	20	20

2. System Description

This system is a basic NetHack game that generates its dungeons with the help of a Twitter API. The system will interact with a player and execute the player's choice/choices. The game will be structured very simply by first opening up a game menu where the player can create his/her character and create a new game or join an already existing game. When the player has entered a game at least one dungeon will appear on the screen in which the player can enter if he/she chooses. When the player enters a dungeon, a monster will appear inside the dungeon. Here the player can choose between several interaction alternatives. The interaction alternatives with the monster are Fight the monster, flee or play poker.

The system that we are constructing will also include a multiplayer mode which will be activated when two or more players are in the same game and in the same dungeon.

The multiplayer mode will contain a dungeon chat where the players that are in the cave can communicate. In the multiplayer mode, the players will be able to interact with each other through the items that the system contains. The item types in the game will be weapons and potions. For example, if a player chooses to use a weapon on another player he will inflict a certain amount of damage on the targeted player. If the player chooses to use potions on another player then the other player will gain health/mana points based on what potion is being used. A player will also be able to use potions on himself to gain health/mana points. The player should be able to exit the game at any point. When the player has chosen to exit the game, the player's character is saved and the game closes.

3. Prioritised List of Use Cases

The ordering of the use cases that we have made are based on the use case's importance to contribute to a minimum viable game.
To be able to construct a minimum viable product, which in our case is a game, we have ranked the importance of the use cases above. From those use cases we have chosen five of them for the first iteration and motivated why they are ordered as they are. These five use cases are considered by us, the far most important use cases that needs to be implemented/created to construct a minimum viable product. Our minimum viable product will be a simplified version of the TwitterNethack game that is being constructed. In the simplified version we still need a player that plays the game, a game that contains the player and a map that contains at least one dungeon. And to actually be able to play the game we need to implement the enter dungeon case and at least one of the several interaction alternatives between the player and the monster inside the dungeon.

Create character - 50 points

- We chose this use case as the most important because there can't be a game

without a player

Create game - 75 points

- This use case represent the game creation and serve as one the most important use cases for running the game

Open game menu - 40 points

- Open game menu work as a menu in which u can either create game and create a character

Enter dungeon - 20 points

- This is the use case which creates the world in which we navigate through and generate also generates monsters.

Fight with monster - 28.5 points

- We choose this as our fifth most important use case because it lets the player fight with the monster.

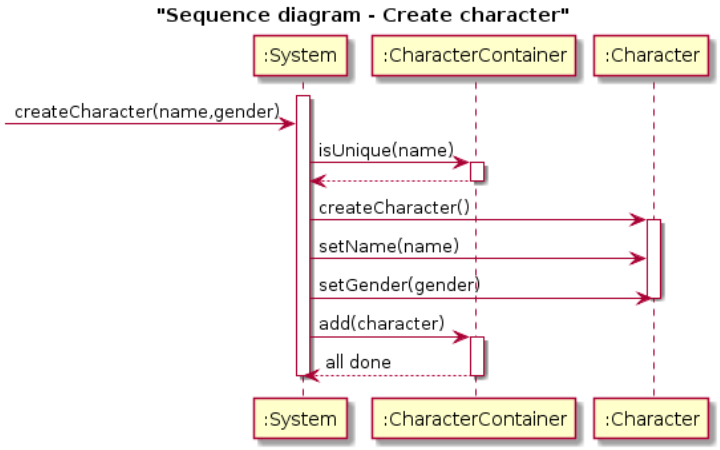
4. System events

1. System Events for Use Case: Create character <Set name and gender for character>
2. System Events for Use Case: Create character <Set race for character>
3. System Events for Use Case: Create character <Set class for character>
4. System Events for Use Case: Create game <Set name for game>
5. System Events for Use Case: Create game <Check if name is taken>
6. System Events for Use Case: Create game <Create game>
7. System Events for Use Case: Enter dungeon <Create dungeon>
8. System Events for Use Case: Enter dungeon <Create monster>
9. System Events for Use Case: Fight monster <Fight monster>
10. System Events for Use Case: Fight monster <Reward player>

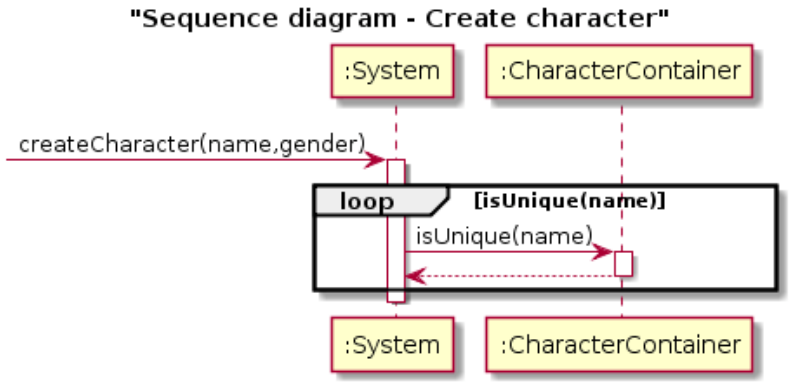
5. Interaction diagrams

5.1 Interaction diagrams for use case: Create character

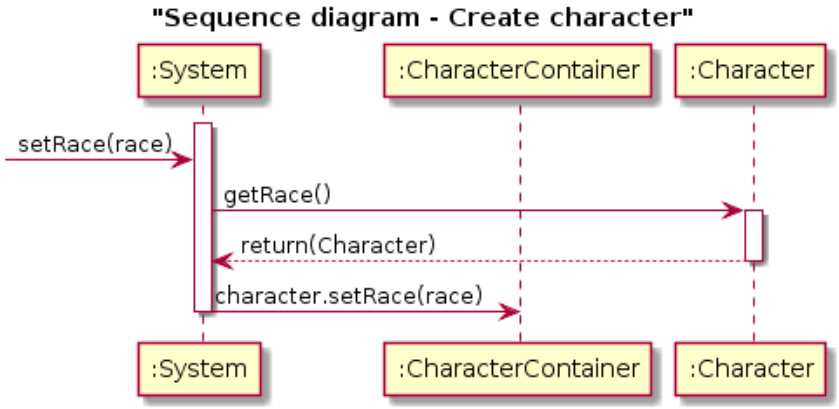
1. Interaction diagram for Use Case: Create character <Set name and gender for character>



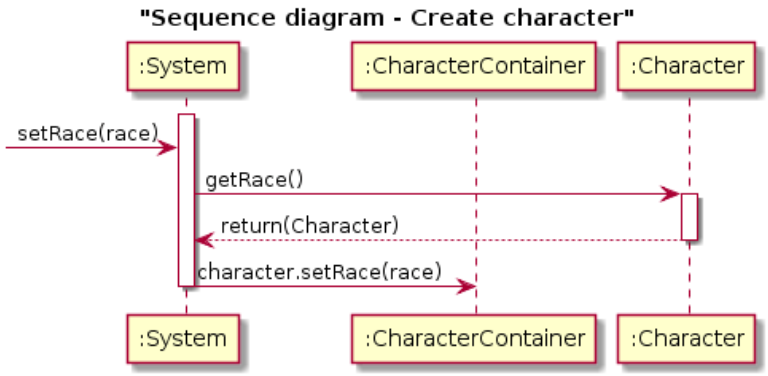
2. Interaction diagram for Use Case: Create character <Check if name is unique>



3. Interaction diagram for Use Case: Create character <Set race for character>

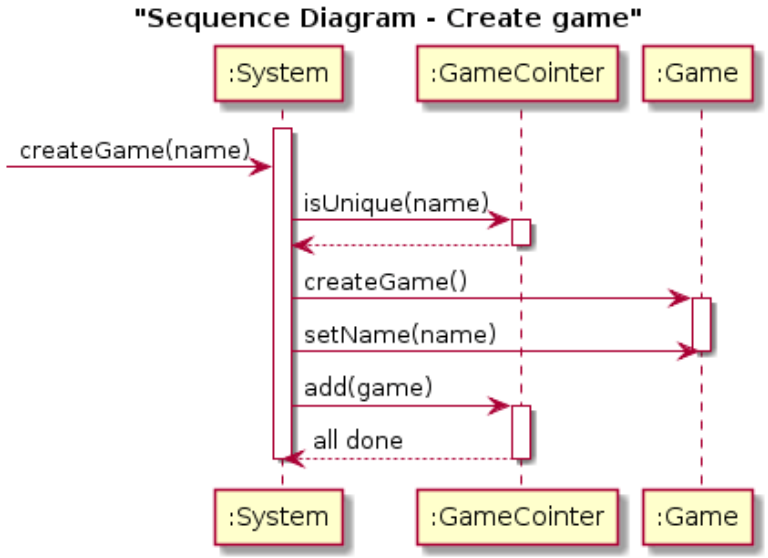


4. Interaction diagram for Use Case: Create character <Set class for character>

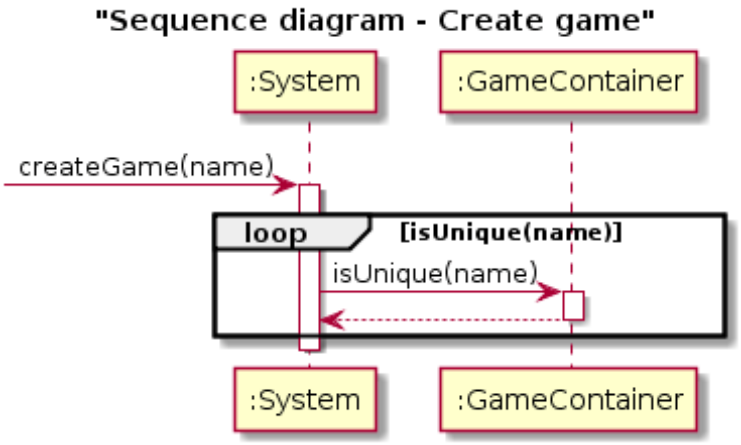


5.2 Interaction diagrams for use case: Create game

5. Interaction diagram for Use Case: Create game <Create game>

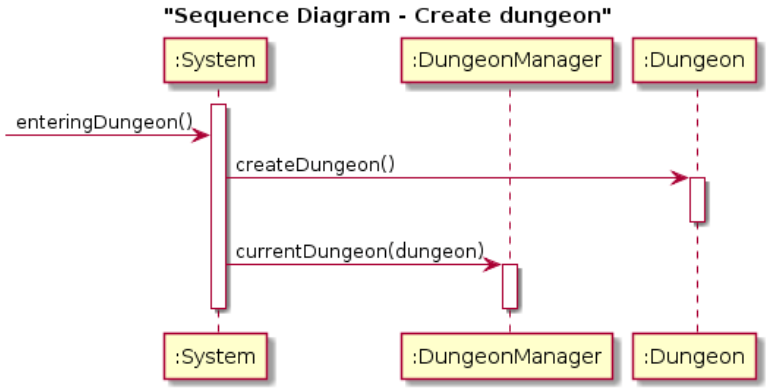


6. Interaction diagram for Use Case: Create game <Check if name is taken>

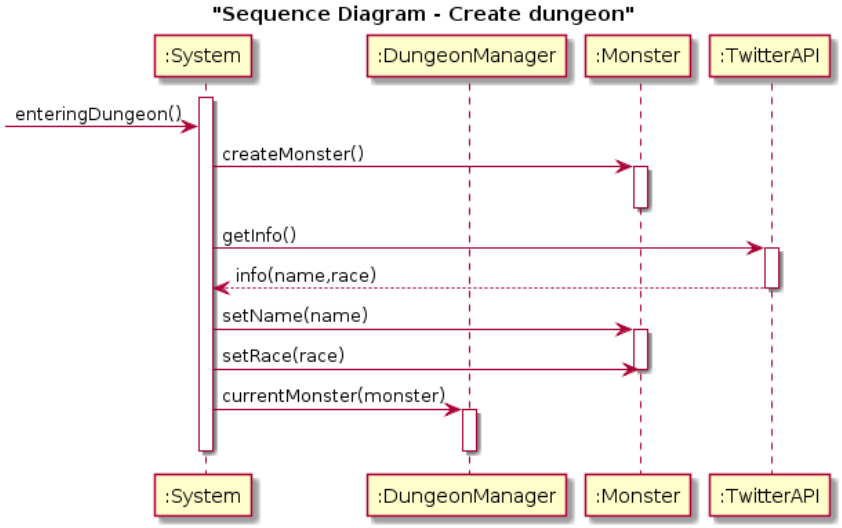


5.3 Interaction diagrams for use case: Enter Dungeon

7. Interaction diagram for Use Case: Enter dungeon <Create dungeon>

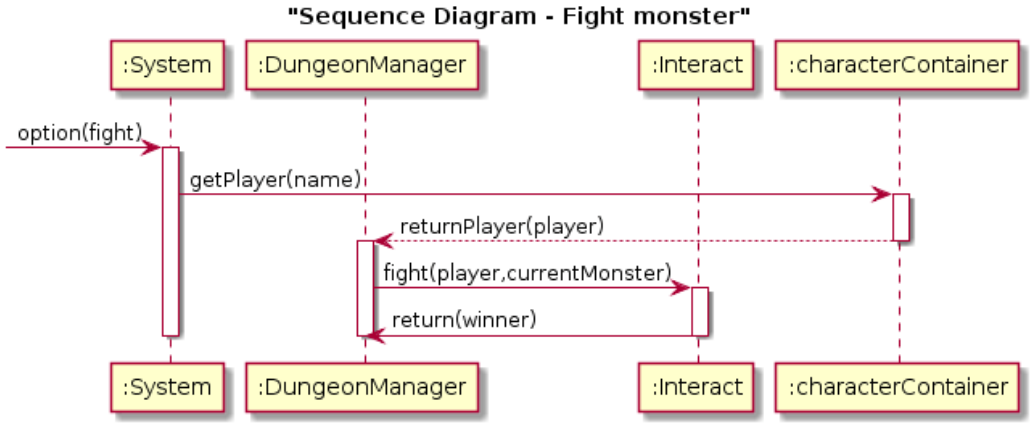


8. Interaction diagram for Use Case: Enter dungeon <Create monster>



5.4 Interaction diagrams for use case: Fight monster

9. Interaction diagram for Use Case: Fight monster <Fight monster>



10. Interaction diagram for Use Case: Fight monster <Reward player>