Detailed Use Cases (Iteration 1) for System TwitterNethack

Assignment in the course PA1435 Object Oriented Design.

April 11th, 2017

Course project on GitHub

Author Name	Social Security Number	Thinking	Writing
Simon Lindgren	960815-1131	25	25
Peter Meunier	951230-0113	10	5
Viktor Enfeldt	960101-9616	20	25
Tobias Fast	961011-6676	25	20
Samuel Asp	960125-1458	20	25

1 System Description

A multi-player dungeon crawler that uses information from Twitter to generate rooms and to populate them with items and characters. The players will be able to freely move around in the room they're in, pick up, drop, equip, craft, and use items, attack monsters and other players, move between rooms in the labyrinth, and chat with other players.

The types of items will range from potions and rocks, to sophisticated weapons like swords and crossbows (the latter of which will require ammunition). It will be possible to craft certain items (such as arrows) by combining two or more other items.

2 Detailed Use Cases

Use case:	**Generate room**
Actors:	REST API (Twitter)
Preconditions:	A seed Twitter user name exists in the system.
Description:	System creates a new room with information from Twitter via the REST API.
Main Success Scenario:	System Generates a room with an exit that is connected to the previous room (unless this is the first room generated) and up to three additional exits mapped to the first three @-mentions on the seed Twitter profile. Room is populated with items and characters based on what system-defined keywords appear in the latest few tweets on the seed Twitter profile.
Extensions:	Seed profile has no @-mentions or keywords in it. System creates an empty room with no other exits apart from the entry one.
Special Requirements:	System has to be able to generate the room very quickly in order to not disrupt the flow of the game.

Use case:	**Create character**
Actors:	Player
Preconditions:	Player started the program and the first room was successfully generated.
Description:	Player selects a character from a preset of options.
Main Success Scenario:	System creates a copy of the preset and gives control over it to the player.
Extensions:	System fails to create a character and program ends.
Special Requirements:	The program needs to create the character within a reasonable amount of time.

Use case:	**Move character**
Actors:	Player
Preconditions:	Player-character exists
Description:	Player requests to move their character in one of eight directions by pressing one of (or a combination of) the WASD keys on their keyboard.
Main Success Scenario:	System moves the character in the requested direction.
Extensions:	There is something blocking the player from moving in that direction (a wall for example). System does not move the character.
Special Requirements:	System must respond quickly to the player's input in order for the game to feel responsive.

Use case:	**Enter room**
Actors:	Player
Preconditions:	Player-Character exists
Description:	Player moves their character into a new adjacent room.
Main Success Scenario:	Use case *Generate Room* is run. System places the character in the newly generated room.
Extensions:	If room is already generated then the system simply places the character in the room
Special Requirements:	System must move the player to the correct position in the new room very quickly in order to not disturb the flow of the game.

Use case:	**Attack Enemy**
Actors:	Player, Enemy
Preconditions:	Player encountered an enemy.
Description:	Player fights a nearby enemy to survive the encounter and gain loot from it.
Main Success Scenario:	Player defeats enemy and survives. Use case *Drop Loot* is run and player can pick up the loot.
Extensions:	Player is either defeated by the enemy and dies, or escapes the fight.
Special Requirements:	It must be visibly obvious how much damage is done by the player.