**2016-2 Object-Oriented Programming Final Project**

김진설 / 전세웅

1. 기초 요구사항 평가

제시된 Object의 기능들을 모두 구현했습니다.

게임의 종료는 메뉴를 이용하거나, 플레이어의 체력이 0이 되었을 때 종료됩니다.

추가적 특이사항은 다음과 같습니다.

Player(@):

지도상에 단 하나만 존재할 수 있습니다. 미리 설정된 기본 공격력과 방어력을 가집니다.

두개 이상의 @이 적혀 있다면 예외로 처리합니다.

Weapon(W), Armor(A):

각각은 지도에 여러 개 존재 가능합니다. 각각의 종류는 단 하나입니다. 무기, 방어구가 이미 있는 상태에서 다른 것을 장비해도 효과가 중첩되지는 않고 1개분의 효과만 적용됩니다. 이때, 새로운 장비는 지도에서 사라집니다.

Monster(M):

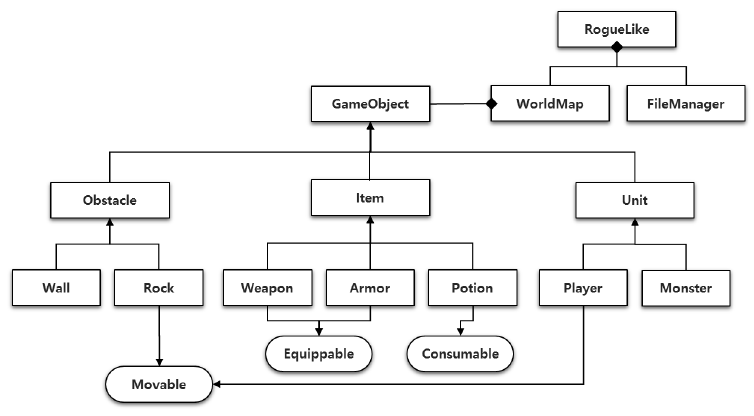
몬스터는 지도에 여러 마리 존재 가능합니다.

몬스터는 기본 공격력과 방어력만 존재합니다.

몬스터를 공격할 때 몬스터에 가하는 데미지 = (플레이어의 기본 공격력 + 무기의 공격력 – 몬스터의 방어력)

먼저 공격한 뒤 몬스터에게 반격을 받습니다.

이때 플레이어가 받는 데미지 = (몬스터의 기본 공격력 - 플레이어의 방어력)



**Config**

Space

각 클래스의 기능은 제시된 기능을 모두 구현하였습니다.

RogueLike: main 메소드를 실행하는 클래스.

WorldMap: 게임 지도를 관리한다. 지도를 초기화하고(initialize), 지도를 화면에 출력하고(printMap), 플레이어를 움직이는(run) 메소드 들을 가진다. 실질적으로 게임을 움직이는 클래스.

FileManager: 파일 입출력을 담당합니다. 최초 실행시 지도를 불러오고(buildMap), 저장(save)과 불러오기(load)를 지원합니다.

GameObject: abstract로 선언된 메소드 showmenu, inputCommand를 가집니다. showmenu는 객체와 플레이어가 상호작용시 가능한 메뉴를 출력하는 역할. inputCommand는 각 객체별로 가능한 상호작용을 구현합니다.

클래스 다이어그램은 전반적으로 제시되었던 것과 동일하지만, 지도에서 ‘마침표’에 해당하는 Space 클래스를 추가했고 각종 상수를 제공하기 위한 Config 클래스가 독립적으로 있습니다. Space는 빈 공간을 표현하는 객체입니다. GameObject를 직접 상속받아 최소한의 필드만 가지고 있으며, 상호작용을 하는 inputCommand 메소드는 player가 Space의 자리로 이동하도록 하는 기능을 가집니다. Config내의 모든 멤버와 메소드는 static으로 선언되어 있으며 플레이어와 몬스터의 최대 체력, 공격력, 방어력, 무기/방어구의 공격력/방어력, 이름, 포션의 회복량, 그리고 Save/Load에 사용되는 파일경로까지 멤버로 가지고, main메소드의 시작과 끝에 출력해줄 화면을 메소드로 가지고 있습니다.

지도 파일의 경우 직사각형 형태가 아닐 때 예외로 간주합니다.

@, #, R, P, M, W, A가 아닌 글자가 map.dat에 있을 시 예외로 간주합니다.

Load시에 저장된 지도의 크기는 map.dat의 지도의 크기와 같아야 합니다.

2. 사용된 OOP개념

사용될 Object들을 구현시에 Inheritance와 Encapsulation을 적용했습니다. 예를 들면, GameObject 클래스에는 좌표와 아이콘만을 멤버로 가지고 getter/setter메소드를 포함하고, 상호작용을 위한 메소드를 abstract로 가지지만, 이것을 상속하는 Unit에서는 추가로 HP와 공격력, 방어력 관련 멤버를 가지고 있으며 그에 따른 getter/setter메소드를 가집니다. 그리고 Unit을 상속하는 Player, Monster 에서는 장비 관련 멤버와 abstract로 되어있는 메소드를 클래스의 목적에 맞게 구현했습니다. 또한 멤버들은 모두 private으로 선언해 직접 접근을 막고 public getter/setter를 이용하도록 했으며 메소드들 중에서도 주된 기능을 하는 메소드를 public으로 설정하고 그러한 메소드들의 동작을 돕기 위한 메소드들은 private으로 설정했습니다.

WorldMap 클래스에서 GameObject들이 저장되는 worldMap 배열에 FileManager에서 읽어온 정보를 각각 객체로 만들어 각 배열의 cell에 넣는 과정에서 업캐스팅이 사용됩니다.

WorldMap 클래스의 move 메소드는 GameObject를 상속하는 와일드카드를 이용한 제네릭 메소드로 구현해 사용했습니다.

클래스의 생성자 구현에서 super()와 this를 사용했고 Movable 인터페이스의 moveTo 메소드 구현에서도 this를 사용했습니다. Config의 멤버와 메소드들은 모두 static으로 선언하여 Config의 객체 생성 없이 바로 사용 가능하도록 하였습니다.

/\*\* 첨부한 테스트 지도 파일 설명

mapBig, mapMedium, mapSmall: 여러 사이즈의 초기 지도파일 입니다. 모든 기능들이 정상적으로 동작합니다.

mapBig\_NoWall: 지도 가장자리에 벽이 없는 경우입니다. 지도의 끝에 도달 시 벽이 있는 것으로 간주합니다.

mapMedium\_2player: 지도에 @이 두 번 적힌 경우입니다. 에러 메세지를 띄우면서 실행이 불가능합니다.

mapWrongShape: 지도가 직사각형이 아닌 경우입니다. 에러 메세지를 띄우면서 실행이 불가능합니다.