CSI2102 Object-Oriented Programming

1. 어플리케이션 설명

화면(WorldMap)상의 Entity는 아래와 같은 종류와 특징을 가진다

@: 플레이어 - WSAD로 상하좌우 한 칸씩 이동. 음식을 먹으면 HP가 상승한다. 단 음식을 많이먹어도 최대 HP이상으로는 상승하지 않는다. 주먹으로 몬스터를 공격할 수 있으며 공격력은 몬스터 보다 낮으므로 무기를 얻어 공격력을 높이고 음식을 먹어 체력을 보충해야 한다. HP가 0이되면 게임 종료

#: 벽 - 플레이어나 다른 객체들이 지나갈 수 없다. 또한 움직이지 않는다

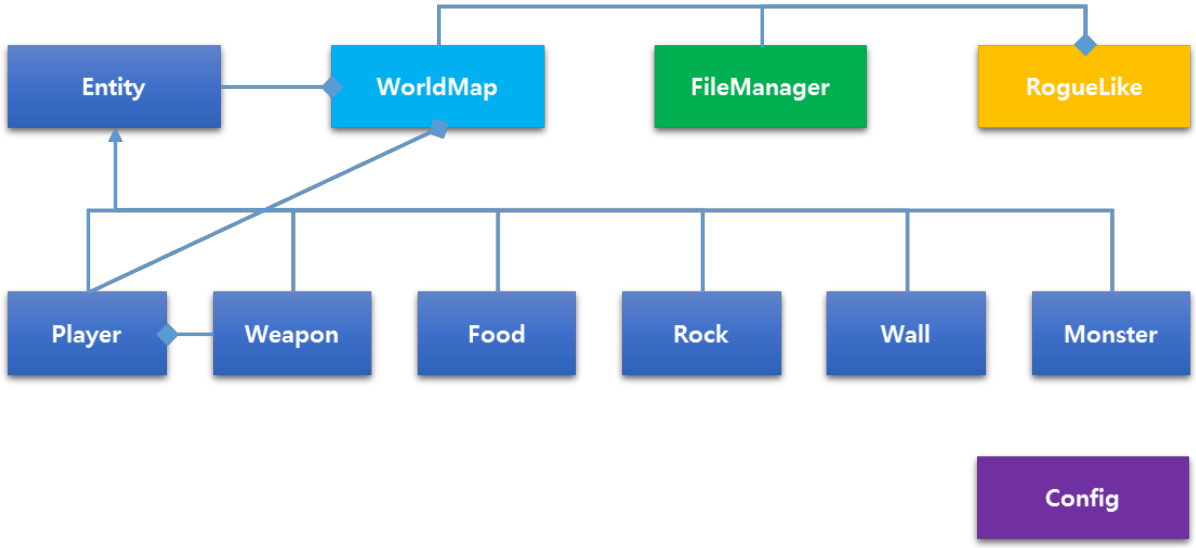
R: 바위 - 플레이어가 한 칸씩 밀 수 있다. 진행 경로에 다른 물체가 있으면 이동시킬 수 없다

F: 음식 - 플레이어가 먹을 수 있으며 음식의 HP만큼 플레이어의 HP가 상승한다.

M: 몬스터 - 움직이지 않으며 플레이어가 다가가도 공격하지 않으나 플레이어가 공격하면 반격한다. HP가 0이 되면 화면에서 사라진다

W: 무기 - 얻으면 자동으로 장비한다.

2. Class 구성



- WorldMap *has-a* Entity
- Food, Rock, Wall, Player, Monster, Weapon *is-a* Entity
- Config는 static으로만 이루어진 class로 독자적으로 사용
- 색상은 서로 다른 패키지에 속한 것을 의미함

RogueLike

- 게임 실행 메인 클래스
- 아래와 같이 구현함

```
public class RogueLike {
	public static void main(String[] args)  {
		WorldMap gameMap = new WorldMap();
		FileManager file = new FileManager();
		file.loadMap(Config.MAP_FILENAME);
		char[][] map = file.buildMap();

		gameMap.initialize(map);
		gameMap.run();

		System.out.println("Game over...");
	}
}
```

Map

Member variable:

- Entity type의 2차원 array map
- Player type의 player 변수 (player의 method를 사용할 일이 많으므로 WorlMap에 멤버로 포함시킴)

Member function:

- insertEntity() : Entity를 map에 삽입
- deleteEntity() : Entity를 map에서 삭제
- initiliaze() : char[][] 배열 데이터에 따라 Entity들을 map에 삽입하여 배치.
- showMap() : 지도를 화면에 출력
- moveEntity() : Entity를 map에서 이동시킴
- showMenu() : 명령 메뉴 출력
- getEnd() : 게임 진행/종료 상태(T/F)를 반환

등…

Entity

Member variable:

- 체력, 최대 체력, 힘, row, col 좌표 등

Member function:

- inputCommand : 명령 입력을 받는 함수(객체 마다 명령이 다름)
- setRow, setCol : Entity의 WorldMap에서의 좌표 저장
- getRow(), getCol() : Entity WorldMap에서의 좌표 반환
- getHP(), getAttackPower() : 체력 및 공격력 반환
- increaseHP(), decreaseHP() : 체력 증가 및 감소 시킴
- getIcon() : 객체의 아이콘(@, #, F 등) 반환
- showMenu() : 객체의 선택 메뉴 출력 (객체 종류마다 출력 내용 다름)

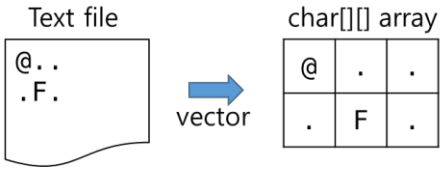
등…

그 외 서브클래스

- Food, Monster, Player, Wall, Rock, Weapon은 Entity를 상속하여 새로운 멤버를 추가하거나 추상 함수를 오버라이딩함

FileManager

- loadMap(): 텍스트 파일(지도 정보)을 한 줄씩 읽어 들여 벡터에 문자열을 한 줄씩 추가
- buildMap(): 벡터의 문자열들을 char[][] 형식의 배열에 저장하여 반환함
- 예)



- 지도 파일은 반드시 c:₩oop₩map.dat 에서 로드할 것

Config

- 전역 변수 및 기호 상수(symbolic constant)들을 정의함
- 모든 출력 문자열 및 문자들은 여기에 상수로 정의하여 사용함
- 입력 관련 메소드를 정의하여 사용함

■ 예)

```
public class Config {
        static Scanner scan = new Scanner(System.in);
        public static int mapRow = 0;
        public static int mapCol = 0;
        final public static char UP = 'W';
        final public static char DOWN = 'S';
        …
        final public static String mainMenu = "(A)Left (D)Right (W)Up (S)Down (Q)uit: ";
        …
```

3. 구현 요구 사항 및 평가

- 각 class는 헤더와 소스파일로 구성되어야 함(class 선언과 정의를 분리)
- OOP 개념이 적용되어야 함(Encapsulation, Polymorphism, Inheritance에 관련된 기능들)
    - Inheritance
    - Encapsulation (private, public, protected, default를 적절히 활용)
    - Up-casting and down-casting
    - Function overloading
    - Template, STL
    - Abstract function
    - Package, constructor, this, static 외 OOP관련 개념 및 기능들
- 컴파일 오류나 런타임 오류가 없어야 함
- 예제 실행파일에 포함된 기능은 모두 만족해야 함(추가 기능 구현 시 Bonus 점수 가능 : 예: 선공형 몬스터, 잠긴 문과 열쇠, 다양한 아이템 구현, 레벨업 등)
- 코드는 같은 코드가 중복되지 않으며 간결하게
- 예외 처리(메뉴에 없는 키 입력 처리 등)를 최대한 고려하여 구현
- 모든 코드는 영문으로 작성

4. 진행 참고사항

- 본 프로젝트는 여러분이 기획, 디자인, 개발 및 테스트까지 담당하는 것입니다. 프로젝트에 필요한 기능은 자신이 결정하여 최선의 것으로 구현 및 완성하시기 바랍니다. 단, 너무 방향성이 없으면 평가가 어려우므로 2~4번과 같은 간략한 정보를 제공하였습니다
- 따라서 "…한 경우에도 ~한 예외 처리를 해야 하나요?", "…클래스(or 메소드 or 변수)가 더 필요한데 추가해도 되나요", "문서에는 ~게 되어 있는데 실제 구현은 …로 해도 되나요" 등의 구현 범위 및 방향에 대한 질문 게시글은 지양하도록 하겠습니다.
- 프로그래밍 언어 레벨에서의 문제점에 대한 질문 글은 가능합니다. 예) ~한 코드에서 NullPointer error가 발생하는데 원인을 모르겠다 등

5. 제출 방법

제출일 : 2016년 6월 6일 오후 6시까지

제출물 :

이클립스 프로젝트 파일을 압축하여 제출 + 보고서
보고서 내용은 class 다이어그램 및 각 class와 기능에 대한 설명, 사용한 OOP 개념, 실행결과 등을 간단하게 작성. 자유형식, 분량제한 없음