|  |
| --- |
| **C++프로그래밍및실습**  **Digital Maze**  **진척 보고서 #2** |

|  |  |
| --- | --- |
|  | 제출일자: 2024.12.01  제출자명: 범지성  제출자학번: 214930 |

**cf) 작성 시점에서 메인 로직에 영향을 주지 않는 코드들은 포함되지 않았을 수 있음**

**1. 프로젝트 목표**

**1) 배경**

퍼즐 게임을 즐겨했기 때문에 간단한 게임에 변형을 주어 비슷한 새로운 게임을 제작하고 싶었음. C++는 내장 GUI 라이브러리가 없어 복잡한 형태의 화면이나 다양한 형태의 이벤트 처리가 어려움. 상하좌우로 움직여 상자를 밀어 모든 상자를 목표물 위에 위치시키는 게임 SOKOBAN은 비교적 형태가 단순하고 이벤트 수도 많지 않았기 때문에 GUI 프로그래밍 없이 CLI 상에서 깔끔한 구현이 용의할 것이라고 판단해 SOKOBAN의 일부 규칙을 채용한 CLI 기반 퍼즐 게임을 기획함.

**2) 프로젝트 목표**

이동횟수가 제한되어 있고 목표지점에 도달하는 퍼즐형 게임을 만드는 것을 목표로함.

**3) 차별점**

게임의 기본적인 형식은 1982년 출시된 “倉庫番(SOKOBAN)”을 참고했고 일부 장애물들은 및 규칙은 이와 비슷한 형식의 퍼즐게임인 “Helltaker”를 모방한다. (플레이어가 한 칸씩 이동, 이동 횟수 제한, 목표지점 도달). 참고한 게임과의 차별점은 다음과 같다

참고작과 달리 게임에 스토리 부여 X

기존 장애물 인부 삭제. 가시 장애물(밟으면 이동횟수 가능 횟수 2회 감소)를 삭제. 레이저 장애물(접촉 즉시 게임 실패) 삭제

새로운 엔티티 추가. 잠긴 벽과 이를 해방하는 열쇠를 추가. 플레이어의 이동에 따라 열리고 닫히기를 반복하는 공간 이동 포탈 추가

튜토리얼 스테이지가 따로 없고 시작화면에서 조작법을 확인할 수 있는 메뉴를 추가한다.

**2. 기능 계획**

**1) 기능 1** 시작 화면

- 설명: 게임의 이름과 스테이지 선택창으로 이동, 게임 설명서로 이동하는 선택지들이 사용자에게 표시되는 화면을 만든다.

(1) 세부 기능 1: 게임 설명서 창

- 설명: 시작화면에서 게임 설명서 선택 시, 아무 키나 누르면 다시 시작화면으로 돌아오는 게임 설명서 창으로 이동한다. 캐릭터 조작법과 엔티티 소개를 담는다.

(2) 세부 기능 2: 선택지 이동 및 확정

- 설명: 상/하 방향키 또는 ‘W’, ‘S’키 입력으로 선택 중인 옵션을 바꾸고 Enter를 눌러 선택을 확정하는 기능을 넣는다.

**2) 기능 2** 스테이지 선택창

- 설명: 전체 스테이지를 띄우고 키보드 입력으로 선택지를 이동하며 선택지를 고르고 Enter로 스테이지를 확정하게 한다. 직전 스테이지를 클리어하지 않으면 다음 스테이지는 잠긴 상태로 둔다. 시작화면으로 돌아가는 키도 구현한다.

(1) 세부 기능 1: 스테이지 로더(Loader)

- 설명: 스테이지 정보를 담은 변수로 게임을 실행하면 스테이지가 1회용이 되고 변수 마다 로직을 붙여줘야 하기 때문에 게임 로직에 사용할 변수를 만들어 거기에 스테이지 정보를 불러오는 것이 좋다. 따라서 사용자의 선택에 따라 스테이지가 로드되는 로직을 만든다.

(2) 세부 기능 2: 선택지 이동 및 확정

- 설명: 상/하/좌/우 방향키 또는 ‘W’, ‘A’, ‘S’, ‘D’키 입력으로 선택 중인 옵션을 바꾸고 Enter를 눌러 선택을 확정하는 기능을 넣는다.

**3) 기능 3** 게임 스테이지 맵

- 설명: 실제 플레이할 게임화면을 구현한다. 현재 스테이지 레벨, 스테이지 맵, 각종 엔티티, 남은 이동 횟수 등을 표기하고, 사용자의 행동에 따라 생긴 변화를 즉시 보여준다.

(1) 세부 기능 1: 스테이지 종료 시 상호작용창

- 설명: 게임 오버 시는 다시하기, 스테이지 선택창으로 이동 선택지를 띄우고 스테이지 클리어 시 두 선택지에 더해 다음 스테이지 플레이하기 선택지를 추가한다.

(2) 세부 기능 2: 스테이지 출력

- 설명: 시작 당시 스테이지를 출력하고 플레이어가 행동을 취할 때마다 현재 출력된 화면을 모두 지운 후 변화된 스테이지와 잔여 이동 횟수를 갱신해서 출력한다.

**4) 기능 4** 클리어 현황 저장소

- 설명: 게임을 종료한 후 다시 실행했을 때 클리어했던 스테이지들을 바로 플레이할 수 있도록 별도의 파일을 조작해 클리어한 스테이지 레벨을 저장하도록 한다.

**5) 기능 5** 플레이어 엔티티

- 설명: 정해진 스테이지 맵 안에서 한 번에 한 칸씩 이동할 수 있고 플레이어의 키보드 입력에 따라 동작한다.

(1) 세부 기능 1: 이동 및 행동

- 설명: 상/하/좌/우 방향키 또는 ‘W’, ‘A’, ‘S’, ‘D’키 입력으로 플레이어가 맵 상에서 이동, 장애물 부수기, 장애물 밀기, 장애물 치기 등의 행동을 한다. 이동 또는 행동을 하면 잔여 이동 횟수가 1 줄어든다.

(2) 세부 기능 2: 행위 유효성 체크

- 설명: 플레이어가 이동하려는 방향에 장애물이 있는지 체크하고 장애물 유무에 따라 플레이어의 행위 여부를 결정한다.

**6) 기능 6** 게임오버 조건 검사

- 설명: 플레이어의 잔여 이동횟수가 0인 상태에서 플레이어가 이동을 시도하면 게임 오버된다.

**7) 기능 7** 기타 엔티티

- 설명: 맵 위에 존재하는 요소들로 플레이어가 해당 엔티티가 존재하는 좌표로 이동을 시도하면 각기 다른 상호작용이 발생한다.

(1) 세부 기능 1: 벽

- 설명: 모든 스테이지에 존재한다. 스테이지 맵이 생성될 때부터 위치가 고정되는 요소. 플레이어가 해당 위치로 이동 시도 시, 아무 일도 발생하지 않으며 이동횟수도 줄어들지 않는다.

(2) 세부 기능 2: 밀 수 있는 장애물

- 설명: 플레이어가 해당 장애물 위치로 이동 시도 시 플레이어의 위치는 변하지 않고, 장애물이 플레이어가 이동하려던 방향으로 한 칸 밀리고, 이동 횟수가 1회 줄어든다. 추가로 장애물이 벽에 가로막힌 경우에도 이동 횟수가 1회 줄어든다.

(3) 세부 기능 3: 부술 수 있는 장애물

- 설명: 해당 장애물의 위치로 플레이어가 이동을 시도하면 플레이어의 위치는 변하지 않고 해당 장애물이 사라지며, 이동 횟수가 1회 줄어든다.

(4) 세부 기능 4: 열쇠와 잠긴 벽

- 설명: 잠긴 벽은 초기에 (1)”벽”처럼 동작한다. 맵에 존재하는 모든 열쇠 엔티티와 접촉 시 열쇠 엔티티와 잠긴 벽 엔티티가 모두 사라진다.

(5) 세부 기능 5: 포탈

- 설명: 스테이지 위에 두 개의 포탈이 존재하고 플레이어가 1회 이동함에 따라 포탈이 활성화되고 비활성화되기를 번갈아 반복한다. 포탈이 활성화되는 동시에 플레이어가 포탈 위치로 이동하면 나머지 한 쪽 포탈이 있던 위치로 이동되며 포탈은 소멸한다. (\*주의\* 비활성화된 포탈 위에 플레이어가 지나갈 때 포탈이 소멸되지 않도록 로직을 만든다.)

(6) 세부 기능 6: 목적지 엔티티

- 설명: 모든 스테이지에 존재한다. 플레이어의 잔여 이동횟수가 0 이상인 상태에서 해당 엔티티 위치로 이동 시 스테이지 클리어. 다음 스테이지가 잠겼다면 다음 스테이지 해금

**8) 기능 8** 실행취소

- 설명: 게임을 진행하던 중 의도하지 않았던 버튼을 눌러서 처음부터 다시 플레이 해야 되는 경우가 생길 수 있기 때문에 게임 플레이 중 “CTRL + z”를 누르면 이전 상태로 되돌리 수 있는 기능을 추가한다.

**9) 기능 9** 세부 스테이지

(1) Stage 1

- 설명: 벽 엔티티들과 하나의 목적지 엔티티로 이루어진 미로형 스테이지

(2) Stage 2

- 설명: 밀 수 있는 장애물과 부술 수 있는 장애물, 벽 엔티티로 목적지 엔티티로 향하는 길을 방해하는 퍼즐형 스테이지

(3) Stage 3

- 설명: 밀 수 있는 장애물, 부술 수 있는 장애물, 열쇠와 잠긴 벽, 벽 엔티티로 목적지 엔티티로 향하는 길을 방해하는 퍼즐형 스테이지

(4) Stage 4

- 설명: 밀 수 있는 장애물, 부술 수 있는 장애물, 열쇠와 잠긴 벽, 포탈, 벽 엔티티로 목적지 엔티티로 향하는 길을 방해하는 퍼즐형 스테이지

**3. 진척사항**

**1) 클래스 및 함수 구현**

**(1) Frame**

- 설명:

화면 디자인 및 출력에 관여하는 함수들의 집합이다.

- public 함수:

(1) void PrintLine(int num)

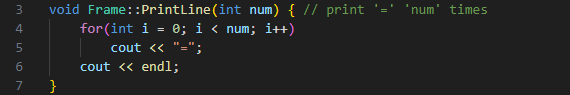
-매개변수

1. num: 출력할 문자의 개수

- 기능:

num의 수 만큼 ‘=’를 출력한 후 줄을 바꾼다.

- 코드



(2) void PrintTitle(int num, string name)

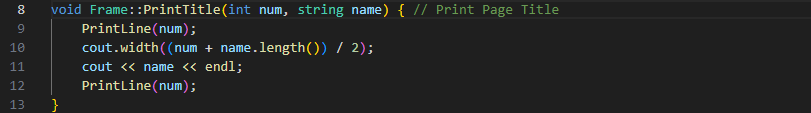
-매개변수

1. num: 제목에 할당될 가로 폭 길이
2. name: 출력할 제목 텍스트

- 기능:

PrintLine()을 호출한 후, 할당된 범위의 중앙에 name을 출력한 후, 다시 PrintLine()을 호출한다.

- 코드



(3) void PrintConfirmAlert()

- 기능:

Enter키로 선택을 확정하라는 안내 문구를 출력한다.

- 코드



(4) void PrintOption(string name, bool selected, int num)

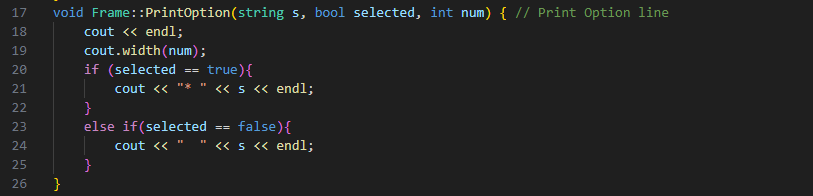
-매개변수

1. name: 출력할 옵션 텍스트
2. selected: 해당 옵션이 선택됐는지 여부
3. num: 출력물 앞 공백 크기

- 기능:

선택된 옵션이면 앞에 ‘\*’을 붙여서 아니면 그냥 앞을 공백으로 두고(여백은 num만큼) name을 출력한다.

- 코드



(5) void PrintButtonLine(int start, int selected)

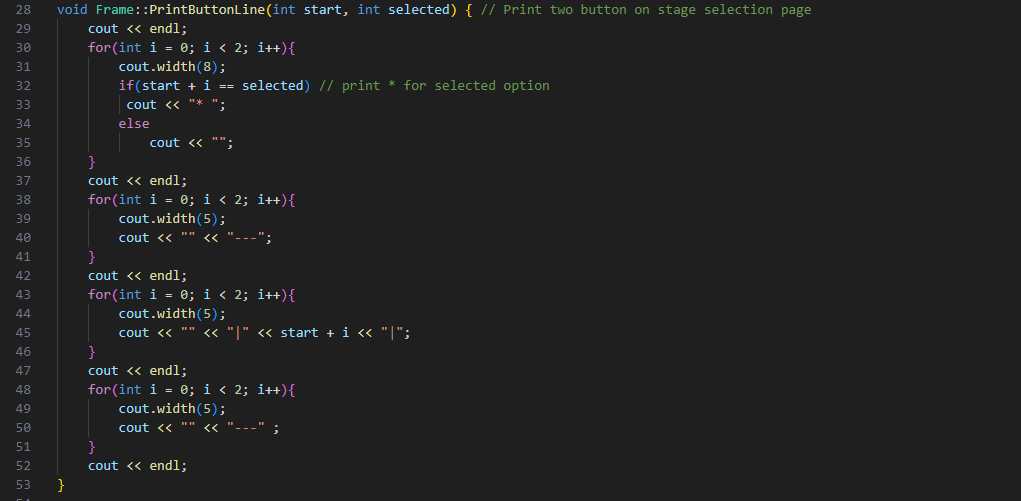
-매개변수

1. start: 출력할 두 버튼 중 첫 버튼의 번호
2. selected: 선택된 버튼의 번호

- 기능:

start번 버튼과 그 다음 순서 버튼을 같은 줄에 출력하고, selected에 해당하는 번호의 버튼이면 버튼 앞에 ‘\*’이 출력된다.

- 코드



(6) void PrintStage(char board[][12], int size)

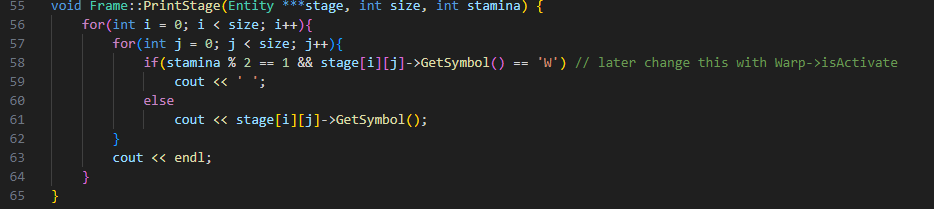
-매개변수

1. board: 출력할 12 x 12 크기의 게임판
2. size: 출력할 게임판의 가로, 세로 길이

- 기능:

게임판을 입력 받아 이를 그대로 출력해준다.

- 코드



**(2) KeyListener**

- *적용된 배운 내용*

정적 멤버를 선언해 객체 생성 없이 클래스만으로 바로 함수와 멤버에 접근한다.

- 설명:

일반적으로 생각하는 listener와 달리 키 이벤트에 따라 특정 작업을 handling하지는 않는다. 보유한 함수를 통해 특정 상황에서 사용자에게 특정 키 입력만 허용한다.

- static 상수(해당 키에 대응하는 아스키 코드 정수값):

1. UP: 위쪽 방향키
2. DOWN: 아래쪽 방향키
3. LEFT: 왼쪽 방향키
4. RIGHT: 오른쪽 방향키
5. ENTER: 엔터키
6. CTRL\_Z: Ctrl + z 키

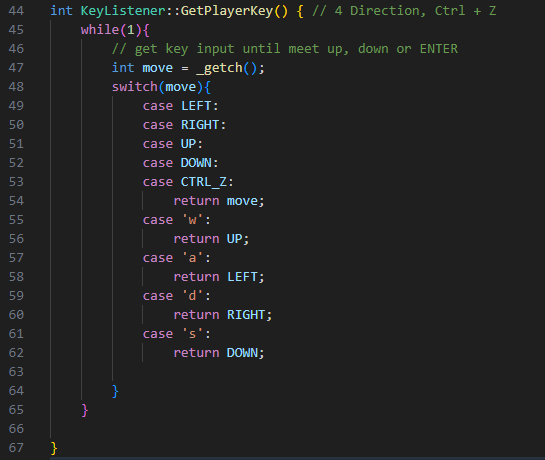
- public static 함수:

(1) int GetPlayerKey()

- 기능:

버퍼 없이 바로 키 하나를 입력 받고, 해당 값이 상/좌/하/우 방향키 또는 w/a/s/d키 일 때 UP/LEFT/DOWN/RIGHT를 반환하고, Ctrl + Z키 일 때는 CTRL\_Z를 반환하며 나머지 키가 입력될 때는 앞서 나온 키 중 하나가 입력될 때까지 다시 키 입력 받기를 시도한다.

- 코드

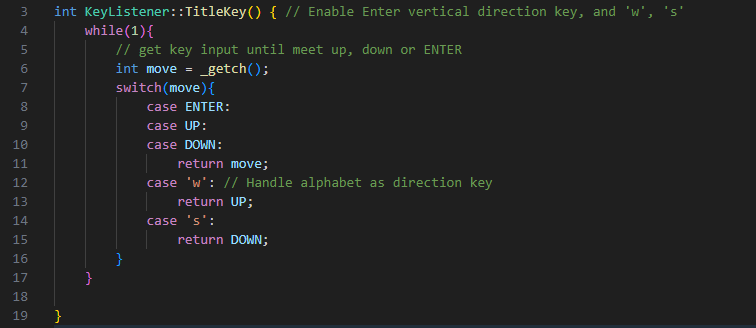


(2) int TitleKey()

- 기능:

버퍼 없이 바로 키 하나를 입력 받고, 해당 값이 상/하 방향키 또는 w/s키 일 때 UP/DOWN를 반환하고, Enter키 일 때는 ENTER를 반환하며 나머지 키가 입력될 때는 앞서 나온 키 중 하나가 입력될 때까지 다시 키 입력 받기를 시도한다.

- 코드

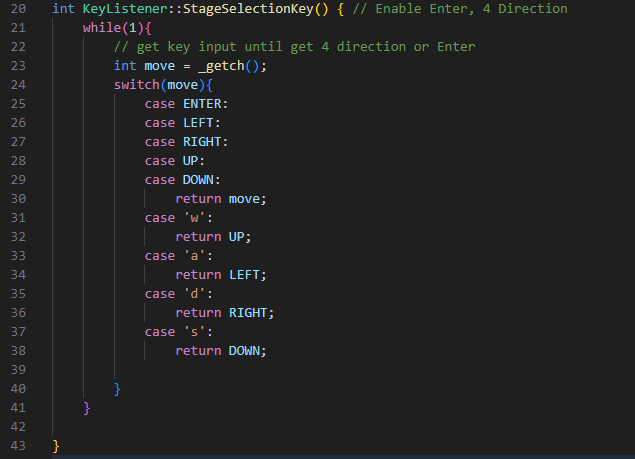
****

(3) int StageSelectionKey()

- 기능:

버퍼 없이 바로 키 하나를 입력 받고, 해당 값이 상/좌/하/우 방향키 또는 w/a/s/d키 일 때 UP/LEFT/DOWN/RIGHT를 반환하고, Enter키 일 때는 ENTER를 반환하며 나머지 키가 입력될 때는 앞서 나온 키 중 하나가 입력될 때까지 다시 키 입력 받기를 시도한다.

- 코드

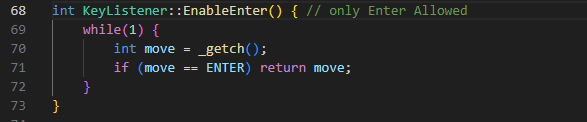


(4) int EnableEnter()

- 기능:

버퍼 없이 바로 키 하나를 입력 받고, 해당 값이 Enter키 일 때 ENTER를 반환하며 나머지 키가 입력될 때는 앞서 나온 키 중 하나가 입력될 때까지 다시 키 입력 받기를 시도한다.

- 코드

****

**(3) Entity**

- *적용된 배운 내용*

인스턴스가 함수를 호출할 때 해당 함수는 인스턴스의 맴버에 변화를 주는 일이 없도록 해야함을 명시하기 위해 함수 선언 시 함수를 const로 선언할 수 있다.

- 설명:

게임 플레이 중 생성되는 모든 엔티티의 공통 분모로 일부 클래스의 상위 클래스다.

- private 필드:

1. symbol: 엔티티의 종류를 구분하는 기호

- 생성자:

(1) Entity()

- 기능:

symbol을 ‘ ‘으로 초기화하며 인스턴스를 생성한다.

- 코드



(2) Entity(char s)

-매개변수

1. s: 생성될 인스턴스가 가지는 기호

-기능:

symbol을 s로 초기화하며 인스턴스를 생성한다.

-코드



- public 함수:

(1) void SetSymbol(char c)

-매개변수

1. c: 인스턴스가 새롭게 가질 기호

- 기능:

인스턴스의 symbol을 c로 변경한다.

- 코드



(2) char const GetSymbol()

- 기능:

인스턴스의 symbol을 반환한다.

- 코드



**(4) Player**

- 설명:

게임 플레이를 할 유저 인스턴스를 생성하기 위한 클래스다. Entity 클래스를 상속하고 있다.

- private 필드:

1. stamina: 유저의 잔여 이동 가능 횟수

- 생성자:

(1) Player()

- 기능:

Player 인스턴스 기본 생성자, stamina가 0인 인스턴스를 생성한다.

- 코드



(2) Player(int s)

-매개변수

1. s: 인스턴스가 가질 스테미나 값

-기능:

stamina값이 s이고 상위 클래스 Entity의 멤버 symbol이 ‘P’인 Player 인스턴스를 생성한다.

-코드



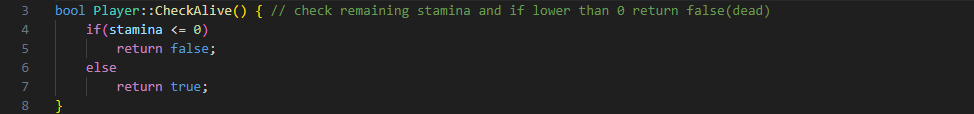
- public 함수:

(1) bool CheckAlive()

- 기능:

인스턴스의 stamina가 0 이하인지 체크해서 0이하면 false 그 외 true를 반환한다.

- 코드



(2) int GetStamina()

- 기능:

인스턴스의 stamina 반환

- 코드



(3) void IncreaseStamina()

- 기능:

객체의 stamina 값을 1 증가한다.

- 코드



(4) void DecreaseStamina()

- 기능:

인스턴스의 stamina가 1 감소한다.

- 코드



**(5) Portal**

- 설명:

Entity 클래스를 상속한 게임 스테이지 상의 구조물에 관한 클래스다. 해당 클래스의 인스턴스는 반드시 한 쌍을 이루어 생성되며, 플레이어가 두 인스턴스 중 한 인스턴스에 접촉하며 나머지 한 쪽으로 순간이동 되고 해당 Portal 인스턴스 쌍은 소멸한다.

- private 필드:

1. x: 자신의 생성 위치 x좌표
2. y: 자신의 생성 위치 y좌표
3. next\_x: 연결된 Portal 인스턴스의 x좌표
4. next\_y: 연결된 Portal 인스턴스의 y좌표

- 생성자:

(1) Portal(int x, int y)

-매개변수

1. x: 인스턴스의 x 좌표
2. y: 인스턴스의 y 좌표

- 기능:

맵 상에서 생성된 위치를 기준으로 x, y 값을 초기화하고 symbol을 ‘W’로 초기화한 Portal 인스턴스를 생성한다.

- 코드

텍스트, 스크린샷, 폰트, 라인이(가) 표시된 사진

자동 생성된 설명

- public 함수:

(1) void Connect(Portal \*p)

-매개변수

1. p: 함수 호출한 인스턴스와 연결할 Portal 인스턴스

- 기능:

p의 x, y 좌표를 Connect를 호출한 인스턴스의 next\_x와 next\_y로 저장한다.

- 코드

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

(2) int GetConnectedX()

- 기능:

인스턴스에 연결된 Portal의 x좌표를 반환한다.

- 코드



(3) int GetConnectedY()

- 기능:

인스턴스에 연결된 Portal의 y좌표를 반환한다.

- 코드



**(6) StageNode**

- 설명:

스테이지 정보, 플레이어 좌표, 플레이어 체력 정보를 담은 인스턴스를 제작하기 위한 클래스다. 게임 플레이 중 실행취소 기능을 구현하기 위한 노드 생성용 클래스

- public 필드:

1. x: 플레이어 x좌표
2. y: 플레이어 y좌표
3. stage: 스테이지 맵 상의 엔티티가 가지고 있던 기호 정보

- public static 상수 필드:

1. SIZE: 스테이지 맵 가로, 세로 폭

- 생성자

(1) StageNode(Entity \*\*\*stage, int x, int y)

-매개변수

1. stage: 스테이지 맵 정보
2. x: 스테이지 상 Player 인스턴스의 열 좌표
3. y: 스테이지 상 Player 인스턴스의 행 좌표

- 기능:

플레이어의 x, y 좌표 정보를 저장하고, stage의 모든 정보가 아닌 기호 정보만 추출해 2차원 배열 형태로 저장한다.

- 코드

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

**(7) Stage**

- *적용된 배운 내용*

객체를 동적 할당 하는 new 키워드와 동적 할당된 메모리의 할당을 해제하는 delete 키워드를 통해 메모리 관리를 한다.

- 설명:

stage map 생성, game play 기능을 가진 클래스다.

- private 상수:

1. SIZE: 맵의 가로 세로 최대 사이즈

- private 필드:

1. stage: 현재 플레이 중인 스테이지 맵의 상태
2. x: Player 인스턴스의 x좌표
3. y: Player 인스턴스의 y좌표
4. stack: stack 인스턴스, 플레이어 행동 직전 상태를 담은 StageNode 객체를 원소로 하는 스택

- 생성자

(1) Stage(int stage\_flag)

- 매개변수:

1. stage\_flag: 스테이지 번호

- 설명

스테이지를 생성하는 함수가 호출되면서 생성되는 인스턴스의 필드들을 초기화한다.

- 코드



- private 함수:

(1) void BuildSupport(int stamina, char stage[][SIZE])

- 매개변수:

1. stamina: 생성할 Player 엔티티의 stamina 변수값
2. stage: 생성할 스테이지 맵의 초기 엔티티 배치 정보가 담긴 배열

- 기능:

stage의 엔티티 위치 정보에 따라 해당 위치에 알맞은 Entity 인스턴스를 동적으로 할당한다. 생성된 한 쌍의 포탈을 연결하는 작업도 이 과정에서 수행된다.

- 코드

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

(2) void BuildStage(int stage\_flag)

-매개변수

1. stage\_flag: 선택된 스테이지 번호

- 기능:

비어있던 stage필드에 배열을 동적으로 할당한다. stage\_flag에 따라 미리 만들어진 스테이지 맵의 배치 정보를 BuildSupport 함수에 보내 할당한 배열에 Entity 인스턴스들을 할당한다.

- 코드

**텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명**

**스크린샷, 텍스트, 라인이(가) 표시된 사진

자동 생성된 설명**

**텍스트, 스크린샷, 멀티미디어 소프트웨어, 디스플레이이(가) 표시된 사진

자동 생성된 설명**

(4) void ChangeBoard(int action)

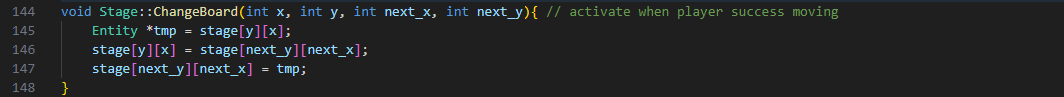
-매개변수

1. x: 위치를 바꿀 엔티티 중 하나의 x좌표
2. y: 위치를 바꿀 엔티티 중 하나의 y좌표
3. next\_x: 나머지 타겟 엔티티의 x좌표
4. next\_y: 나머지 타겟 엔티티의 y좌표

- 기능:

좌표를 받은 두 엔티티의 stage 상의 위치를 swap한다.

- 코드



(4) void Warp(int next\_x, int next\_y)

-매개변수

1. next\_x: 플레이어가 접근한 Portal 인스턴스의 x좌표
2. next\_y: 플레이어가 접근한 Portal 인스턴스의 y좌표

- 기능:

인자로 주어진 좌표 상에 존재하는 Portal 인스턴스와 연결된 Portal 인스턴스의 위치로 Player 인스턴스의 위치를 옮기고 여기에 사용된 한 쌍의 Portal 인스턴스의 기호를 ‘ ‘로 변경해 포탈로써의 기능을 무력화한다.

- 코드

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

(5) void Undo(Player \*user)

-매개변수

1. user: 현재 게임을 플레이 중인 Player 인스턴스의 참조변수

- 기능:

이전 행동을 취소하는 동작이다. user의 스테미나를 1 증가하고, stack의 최상단에 있는 StageNode 인스턴스를 가져와 user가 참조하는 Player 인스턴스를 StageNode에 저장된 직전 유저의 위치로 이동시키고 엔티티 파괴나 엔티티 이동에 의한 변화를 이전 상태로 복구하기 위해 StageNode 속 이전 맵의 정보를 토대로 Entity들의 symbol을 재설정한다. 이전 상태 복구가 끝난 후 사용된 StageNode는 stack에서 버려진다.

- 코드

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

(6) void Unlock()

- 기능:

해당 함수가 호출되면 Lock 엔티티 역할을 하고 있던 Entity 인스턴스들의 기호를 모두 ‘ ‘로 변경해 Lock 엔티티를 무력화한다.

- 코드

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

- public 함수:

(1) int Play(Frame f, int stage\_flag)

-매개변수

1. f: Frame 인스턴스
2. stage\_flag: 선택된 스테이지 레벨

- 기능:

스테이지를 플레이한다. 스테이지 레벨, 스테이지 맵 현황, 남은 스테미나가 출력된다. 플레이어는 이동 또는 실행 취소를 할 수 있다. 플레이어가 이동을 할 때마다 행동 직전의 상태를 임시 저장한다. 실행 취소를 하면 저장했던 직전 상태를 불러온다. 행동이 끝나면 해당 행동을 반영해서 화면을 재출력한다. 남은 이동횟수가 0이 됐을 때 이동을 시도하면 Game over (0 반환), 이동횟수 안에 목적지에 도달하면 Game Clear (1 반환), 플레이어 행동에 대한 자세한 서술은 기능 구현에서 한다.

- 코드

스크린샷, 텍스트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

스크린샷, 텍스트, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

**2) 기능 구현**

**(1) 시작화면**

- 변수

1. selected: 시작화면에 제공된 선택지 중 선택된 것의 인덱스를 true로 하는 변수
2. move: 사용자로부터 받은 입력을 저장하는 변수
3. option: 현재 선택된 옵션을 가리키는 포인터
4. page\_flag: main함수에서 실행할 페이지를 가리키는 변수
5. printer: 화면 구성을 위한 Frame 인스턴스

- 설명:

해당 코드는 사용자 입력 한 번이 한 iteration인 반복문 속에 있다. system(“cls”)로 현재 화면에 출력된 문자를 모두 지우고 사용자 동작에 따라 변화된 상태를 새로 출력한다. 게임의 제목, 스테이지 선택하기, 게임 설명서, 게임 종료하기 옵션이 있음을 알리기 위해 출력되고, 선택 확정을 Enter로 수행함을 알리는 문구가 출력된다. selected[k]가 true인 경우 해당 옵션은 앞에 ‘\*’ 문자를 붙여 출력된다.

- 코드 스크린샷

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

-세부 기능

1) 설명서 페이지

- 설명:

사용자에게 해당 게임의 조작법, 요소들에 대해 소개하는 페이지를 출력하고 유저의 Enter 입력을 대기한다. Enter 입력이 발생하면 Title로 돌아간다.

-코드

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

**(2) 스테이지 선택창**

- 변수

1. selected: 사용자의 입력을 저장한 변수
2. page\_flag: main함수에서 실행할 페이지를 가리키는 변수
3. printer: 화면 구성을 위한 Frame 인스턴스
4. stage\_flag: 현재 선택된 스테이지 번호를 저장하는 변수

- 설명:

페이지의 제목이 출력된 후, 네 개의 스테이지를 선택할 수 있는 선택지와 시작화면(Title)로 돌아갈 수 있는 선택지가 주어지며, 해당 선택지는 상/좌/하/우 방향키 또는 w/a/s/d 키를 눌러 바꿀 수 있고 Enter로 선택을 확정할 수 있다.

- 적용된 배운 내용

플레이어가 적절한 행동을 취할 때까지 해당 페이지를 계속 실행할 수 있도록 while(true)로 무한 루프를 만들고 if문으로 조건부로 while을 탈출한다.

- 코드 스크린샷

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

**(3) 스테이지 맵**

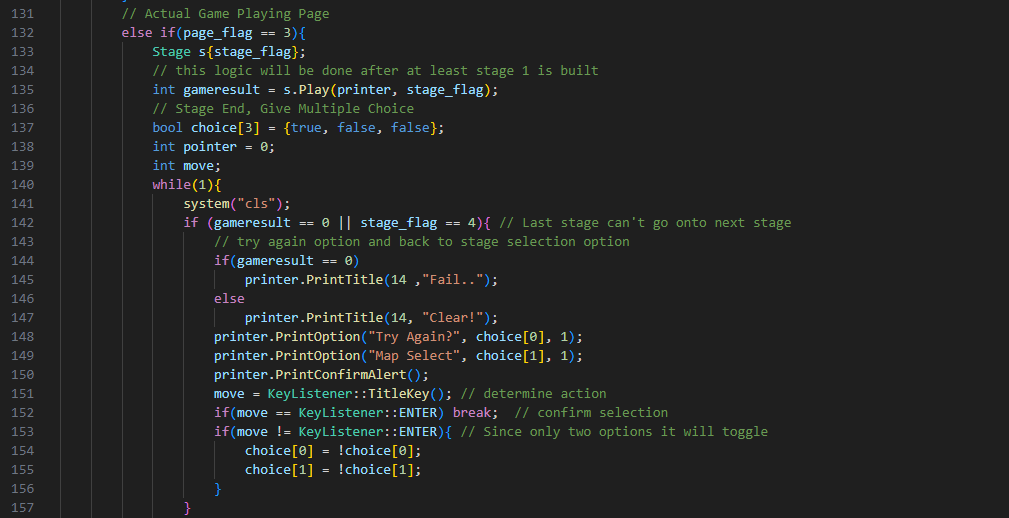
- 변수

1. page\_flag: main함수에서 실행할 페이지를 가리키는 변수
2. printer: 화면 구성을 위한 Frame 인스턴스
3. stage\_flag: 현재 선택된 스테이지 번호를 저장하는 변수
4. gameresult: 게임 결과값을 저장하는 변수
5. choice: 게임이 끝난 후 나온 선택지 중 선택된 것을 표시하는 변수
6. pointer: choice에서 선택된 옵션의 인덱스 포인터 변수
7. move: 유저의 입력을 저장하는 변수

- 설명:

stage\_flag에 따라 Stage 인스턴스를 생성하고 인스턴스의 play() 함수로 게임을 플레이한다. 플레이 결과에 따라 실패 또는 성공 여부를 알리고, 실패했을 때는 다시하기, 스테이지 선택으로 돌아가기 선택지를 유저에게 제공하고, 성공했을 때는 다음 스테이지 플레이하기 선택지를 추가로 보여준다. 단, 마지막 스테이지는 다음 스테이지가 없기 때문에 다음 스테이지 플레이 버튼을 출력하지 않는다.

- 코드 스크린샷



텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

**(5) 플레이어 엔티티**

- 설명:

스테이지 맵 안에서 유저 입력에 따라 한 칸 이동하거나 특정 행동을 취할 수 있는 엔티티 (기호 : ‘P’), 자세한 설명은 Player 클래스 참조

- 세부 기능:

[1] 이동 및 행동

- 설명:

유저의는 사용자의 입력에 따라 이동, 장애물 파괴, 장애물 밀기, 장애물 차기 등의 행동을 수행할 수 있다. 행동을 수행하기 전 현재 상태를 담은 StageNode를 생성해 stack에 push 한다. 각각의 행동은 스테미나를 1 소모한다. 상세한 엔티티별 상호작용은 기타 엔티티에서 후술. 자세한 logic은 Stage 클래스의 함수들 참조

- 코드 스크린샷

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

[2] 행위 유효성 체크

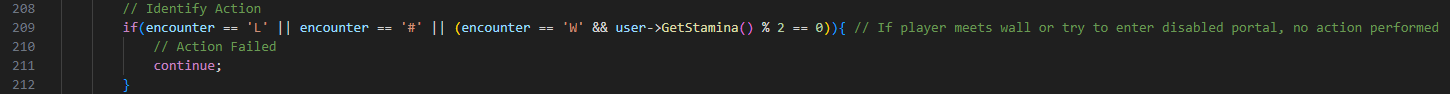
- 변수:

1. encounter: 유저가 향하려는 방향에 자리 잡은 엔티티 기호 저장 변수
2. stage: 스테이지 맵 배열
3. user: Player 인스턴스
4. action: 입력 받은 키 값 저장 변수

- 설명:

사용자가 방향키 입력을 받아 이동을 시도할 때, 유저가 향하려는 방향에 존재하는 엔티티가 무엇인지 판별해 행위 여부를 결정한다. 한 장소에 고정된 파괴 불가능한 엔티티와 접촉하면 그 외 모든 다른 작업 없이 iteration이 종료돼 행동에 실패하게 된다. 행위에 성공하면 바로 위에서 서술한 코드가 실행된다.

- 코드 스크린샷 (주어진 코드는 바로 위 코드와 연결된 if문)



**(6) 게임 오버 조건 검사**

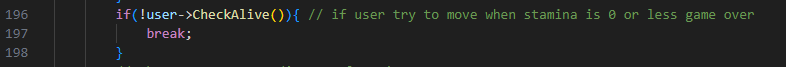
- 변수

1. user: Player 인스턴스

- 설명:

Stage의 play 내부 로직 중 하나. 유저의 잔여 체력을 검사해 체력이 0 이하이면 play를 루프하는 while문을 탈출한다. CheckAlive()는 Player 클래스 참조

- 코드 스크린샷

****

**(7) 기타 엔티티**

[1] 벽(기호 : ‘#’)

- 설명:

스테이지에 고정된 엔티티, 플레이어가 해당 엔티티의 위치로 이동을 시도하면 플레이어가 아무 행동도 하지 않은 것으로 취급된다.

[2] 부술 수 있는 장애물(기호 : ‘B’)

플레이어의 이동을 방해하는 엔티티, 해당 위치로 플레이어가 이동을 시도하면 플레이어의 위치는 변하지 않고 엔티티는 파괴되어 사라진다.

- 코드



[3] 밀 수 있는 장애물(기호 : ‘O’)

플레이어의 이동을 방해하는 엔티티, 해당 위치로 플레이어가 이동을 시도하면 플레이어는 해당 장애물을 찬다(스테미나 1 소모). 플레이어가 해당 장애물을 찼을 때, 장애물 뒤가 빈 공간이면 장애물이 밀려나고 아니라면 플레이어의 스테미나만 소모된 채로 아무 일도 일어나지 않는다.

- 코드

텍스트, 멀티미디어 소프트웨어, 소프트웨어, 스크린샷이(가) 표시된 사진

자동 생성된 설명

[4] 잠긴 벽(기호 : ‘L’)

기본적으로 ‘#’와 동일하게 동작하고 특수한 조건을 만족하면 소멸한다.

[5] 열쇠(기호 : ‘K’)

해당 엔티티가 존재하는 위치로 플레이어가 이동하면 해당 엔티티는 소멸하고 동시에 스테이지에 존재하는 모든 ‘L’ 엔티티도 소멸한다. 상세한 로직은 Stage 클래스의 Unlock() 함수 참조

- 코드

텍스트, 스크린샷, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

[6] 목적지 엔티티(기호 : ‘@’)

플레이어가 해당 엔티티가 있는 위치로 이동하면 스테이지가 클리어를 판별하는 변수가 클리어 했을 때의 값으로 바뀌며 해당 스테이지가 종료된다.

스크린샷, 텍스트, 폰트이(가) 표시된 사진

자동 생성된 설명

[7] 포탈 엔티티(기호 : ‘W’)

플레이어의 잔여 스테미나가 짝수일 때만 활성화되고 활성화된 상태에서는 벽처럼 작동한다. 포탈 엔티티는 두 개가 하나의 쌍을 이루고 있으며 플레이어가 포탈 엔티티가 있는 곳의 좌표로 이동하는데 성공하면 연결된 다른 한 쪽 포탈 엔티티의 위치로 이동하고 포탈 엔티티 쌍은 소멸하게 된다. 자세한 로직은 Stage 클래스 Warp() 함수 참조

- 코드



**(8) 실행 취소**

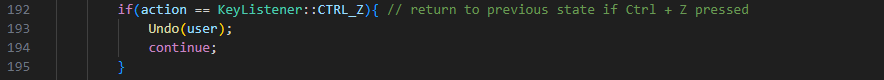
- 변수

1. user: Player 인스턴스
2. action: 유저의 키 입력

- 설명:

사용자가 Ctrl + z를 입력할 때, stack이 비어 있지 않은 경우 스택 최상위에 저장된 직전 스테이지 상태를 불러와 상태를 되돌린다. 자세한 로직은 Stage 클래스 Undo() 참조

- 코드 스크린샷



**(9) 세부 스테이지**

- 설명:

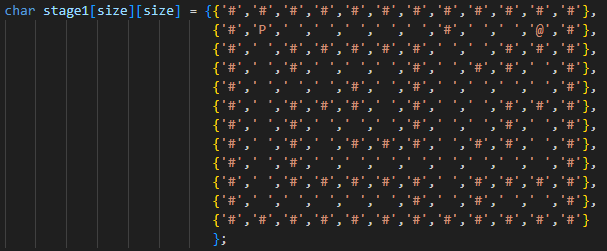
스테이지가 올라감에 따라 스테이지의 엔티티 배치, 엔티티 종류가 달라지기 때문에 스테이지를 생성하는 로직을 만들기보다는 스테이지 모양을 그대로 2차원 배열에 저장한다.

[1] 스테이지 1

- 설명:

벽 엔티티와 목적지 엔티티만 존재하는 미로탈출형 스테이지 해당 스테이지의 유저 stamina는 35다.

-코드

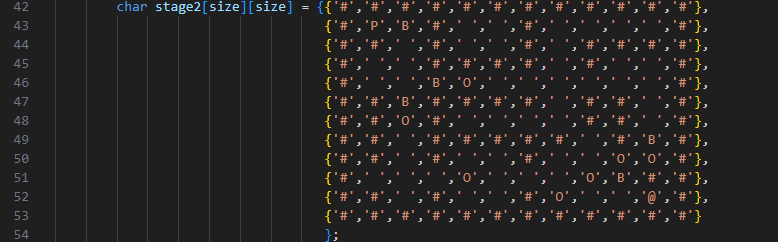


[2] 스테이지 2

- 설명:

벽 엔티티, 목적지 엔티티, 부술 수 있는 엔티티, 밀 수 있는 엔티티로 구성된 스테이지, 유저 초기 stamina는 35다.

-코드



[3] 스테이지 3

- 설명:

벽 엔티티, 목적지 엔티티, 부술 수 있는 엔티티, 밀 수 있는 엔티티로 구성된 스테이지, 유저 초기 stamina는 35다.

-코드

[4] 스테이지 4

- 설명:

벽 엔티티, 목적지 엔티티, 부술 수 있는 엔티티, 밀 수 있는 엔티티로 구성된 스테이지, 유저 초기 stamina는 35다.

-코드

**2) 테스트 결과**

**(1) 시작화면**

- 설명

게임 제목과 옵션, 옵션 포인터 역할인 ‘\*’가 잘 출력됨을 확인, 상/하 방향키 입력 시 선택된 옵션 줄로 ‘\*’ 위치가 이동함을 확인함. 각 옵션 별로 Enter 입력 시 의도한 동작 수행함.

- 테스트 결과 스크린샷

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

- 세부기능:

[1] 페이지 설명서

- 설명:

시작화면의 Instructions 옵션을 선택하면 출력되는 게임 플레이 방법과 게임 엔티티 소개가 담긴 페이지, Enter를 입력하면 시작 화면으로 복귀한다.

- 테스트 결과 스크린샷

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

**(2) 스테이지 선택창**

- 설명

페이지의 제목, 4개의 스테이지 버튼, 시작화면 복귀 옵션, 그리고 선택된 옵션에 대한 ‘\*’ 포인터가 잘 작동했고, 키 입력에 의한 포인터 이동, Enter로 선택 확정이 잘 작동했다.

- 테스트 결과 스크린샷

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

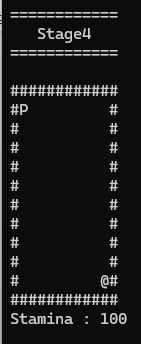
**(3) 스테이지 맵**

- 설명

스테이지 번호가 담긴 제목, 현재 스테이지 맵 상태, 유저의 잔여 체력이 출력됐다.

- 테스트 결과 스크린샷

[1] 게임 플레이



[2] 게임 오버

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

[3] 게임 클리어 (Stage 1~3)

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

[4] 게임 클리어 (최종 Stage)

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

**(5) 플레이어 엔티티**

[1] 이동

- 설명

이동을 시도한 방향으로 맵 상의 위치가 바뀌고 stamina가 1 감소함을 확인, 이동 유효성 체크는 별도의 메시지 등이 없어 문서상 증빙이 불가능 하나 실행 취소 기능을 통해 정상 작동했음을 확인함. (이전 상태 스택에는 이동을 성공했을 때만 노드가 추가되도록 설계했고, 한 번 이동 후, 이동 불가능한 동작을 5번 시도한 후 Ctrl + Z를 한 번 누른 결과, 즉시 최초 위치로 복귀했음을 확인함)

- 테스트 결과 스크린샷

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

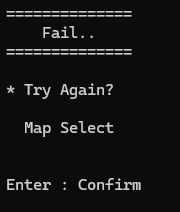
자동 생성된 설명우측 이동 후

**(6) 게임 오버 조건 검사**

- 설명

Stamina가 0일 때 아무 방향으로 플레이어가 이동을 시도했을 때 Game over 상태로 넘어감을 확인함

- 테스트 결과 스크린샷

방향키 입력 후

**(8) 실행 취소**

- 설명

플레이어가 이동을 수행한 상태에서 Ctrl + Z를 누르면 직전 상태로 되돌아간다. 이동 불가능한 위치로 이동을 시도한 경우는 저장되지 않음도 테스트를 통해 확인함(기능 테스트 5-1 플레이어 엔티티 이동 참조)

- 테스트 결과 스크린샷

 Ctrl + Z 입력 이후 텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

**(9) 세부 스테이지**

- 설명:

스테이지 번호 순서대로 나열됨

텍스트, 폰트, 스크린샷, 메뉴이(가) 표시된 사진

자동 생성된 설명

**4. 1차 보고 대비 변경 사항**

**1) Dummy Stage 삭제**

- 이전: 미구현 스테이지 Dummy Stage로 대체

- 이후: Dummy Stage 로직 삭제

- 사유: 모든 스테이지 구현 완료

**2) Moveable 클래스 삭제**

- 사유: 게임 스테이지 맵을 저장하는 변수를 문자열 배열에서 객체 배열로 전환함에 따라, 플레이에 따라 위치가 변하는 엔티티 대부분이 위치 좌표를 기억하고 있을 필요성이 사라짐에 따라 클래스를 삭제함.

**5. 프로젝트 일정**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **업무** | | **11/3** | **11/10** | | **11/17** | | | **11/21** | | **…** |
| **제안서 작성** | | **완료** |  | |  | | |  | |  |
| **기능1, 기능2, 기능3** | |  | **완료** | |  | | |  | |  |
| **기능6, 기능8** | |  | **완료** |  | | |  | | |  |
| **기능 5** | |  | **진행 중(기능 7과 함께 완성 예정)** | | | | | | |  |
| **기능4, 기능7** | |  |  | | **------------------------------------->** | | | | | |
| **기능7 - 1** | |  | **완료** | |  |  | | |  | |
| **기능9** | **Stage1** |  | **완료** | |  |  | | |  | |
|  | **Stage 2** |  |  | | **완료** |  | | |  | |
|  | **Stage3, 4** |  |  | |  | **완료** | | |  | |
| **보수 및 최적화** | |  | **---------------------------------------------------->** | | | | | | | |