게임시스템디자인

스트레인저

PROJECT PRESENTATION

202010967 게임전공 조예진
202111087 게임전공 유지석
202115072 게임전공 서태린
202115073 게임전공 유지효

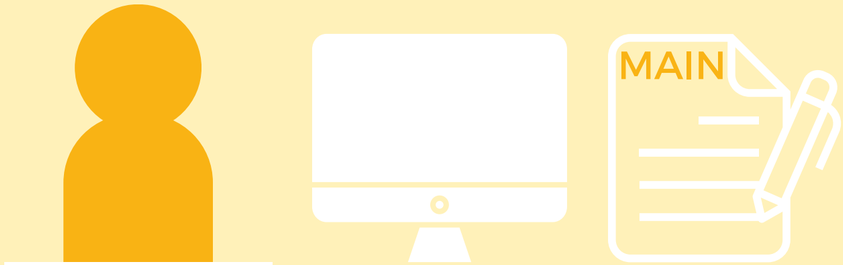
목차

프로젝트 소개 - 팀원

조예진

유지석

서태린



유지효

프로젝트 소개 - 게임

C++

PC

키보드 / 모니터

SDL2

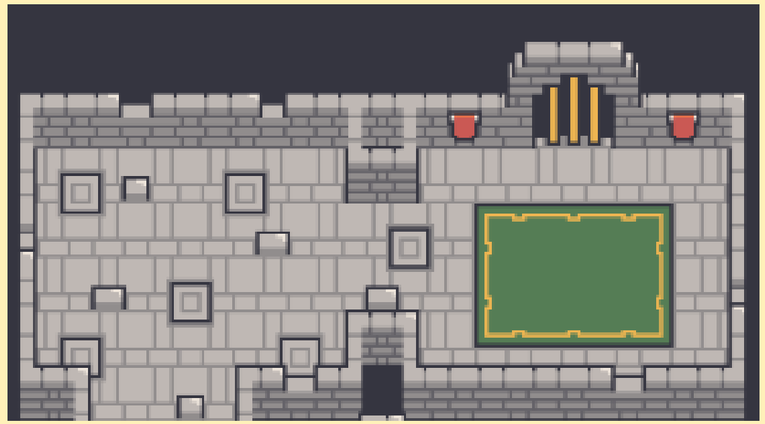
프로젝트 소개 - 게임

초안

장르: 좀비 아포칼립스
목표: 생존자 구조, 탈출
맵=그래픽, 복층
스크립트=콘솔

완성안

장르: 판타지
목표: 탑 클리어, 탈출
맵 및 스크립트=그래픽, 단층



스프라이트 출력
타일 형식 에셋

SDL2 라이브러리

프로젝트 소개 - 게임

공간: 빙고맵

플레이어가 움직이며
진행 가능한 공간

각 칸의 슬라임 퇴치시
벽 해제, 이동 가능

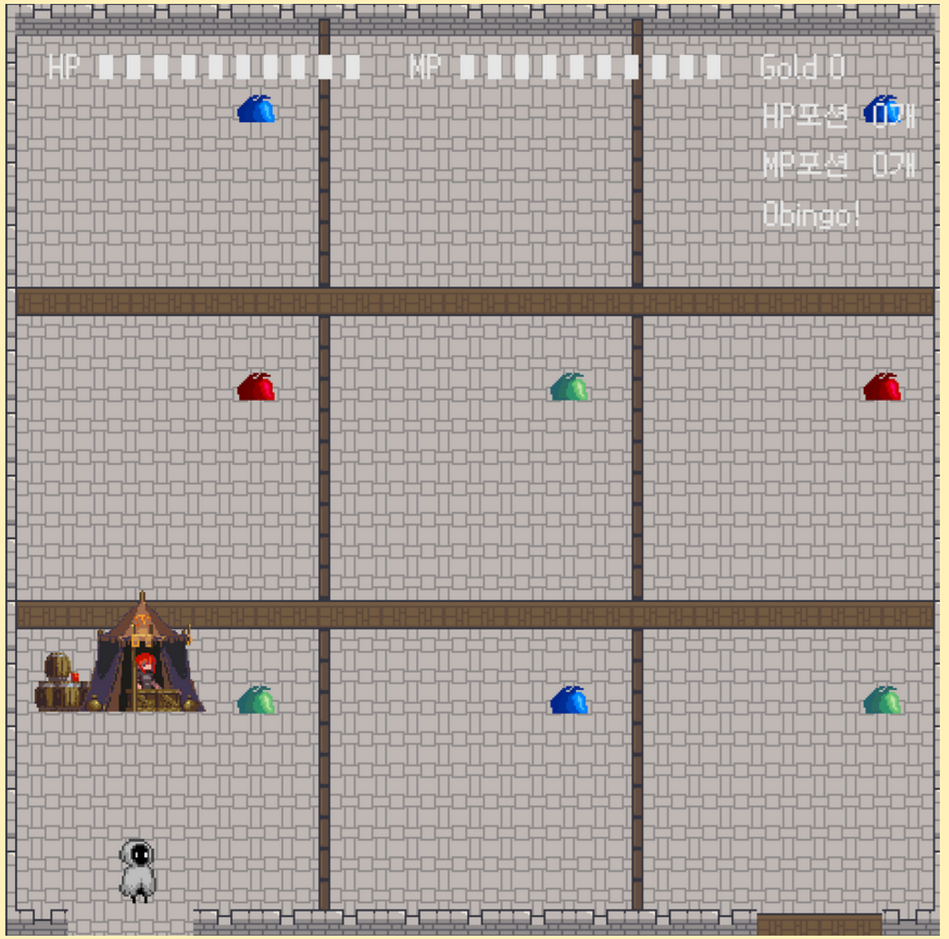
3 bingo 완성시 중앙에
보스 소환

모험가

플레이어가
움직이는 캐릭터

망토 속성 반영

가능한 액션: 이동



상태UI

HP와 MP, 소지 골드,
소지 포션 현황 반영

슬라임

전투맵으로 변경,
전투 승리시 골드 드랍,
막힌 벽 해제

상점

획득한 골드로
필요한 물건 구매 가능

심볼 인카운터 방식 상호작용
상점 물건은 키입력으로 구매

프로젝트 소개 - 게임

공간: 전투맵

플레이어가 몬스터와
턴제 전투를 하는 공간

적 체력이 0 이하가 되면
클리어 판정, 빙고맵으로
돌아간다

모험가

플레이어의 키입력을
통해 전투한다

가능한 행동
-공격 및 스킬 사용
-망토 탈착용
-포션 사용

슬라임

속성 존재
랜덤 체력, 랜덤 공격력 부여

행동: 플레이어를 공격함

프로젝트 소개 - 게임

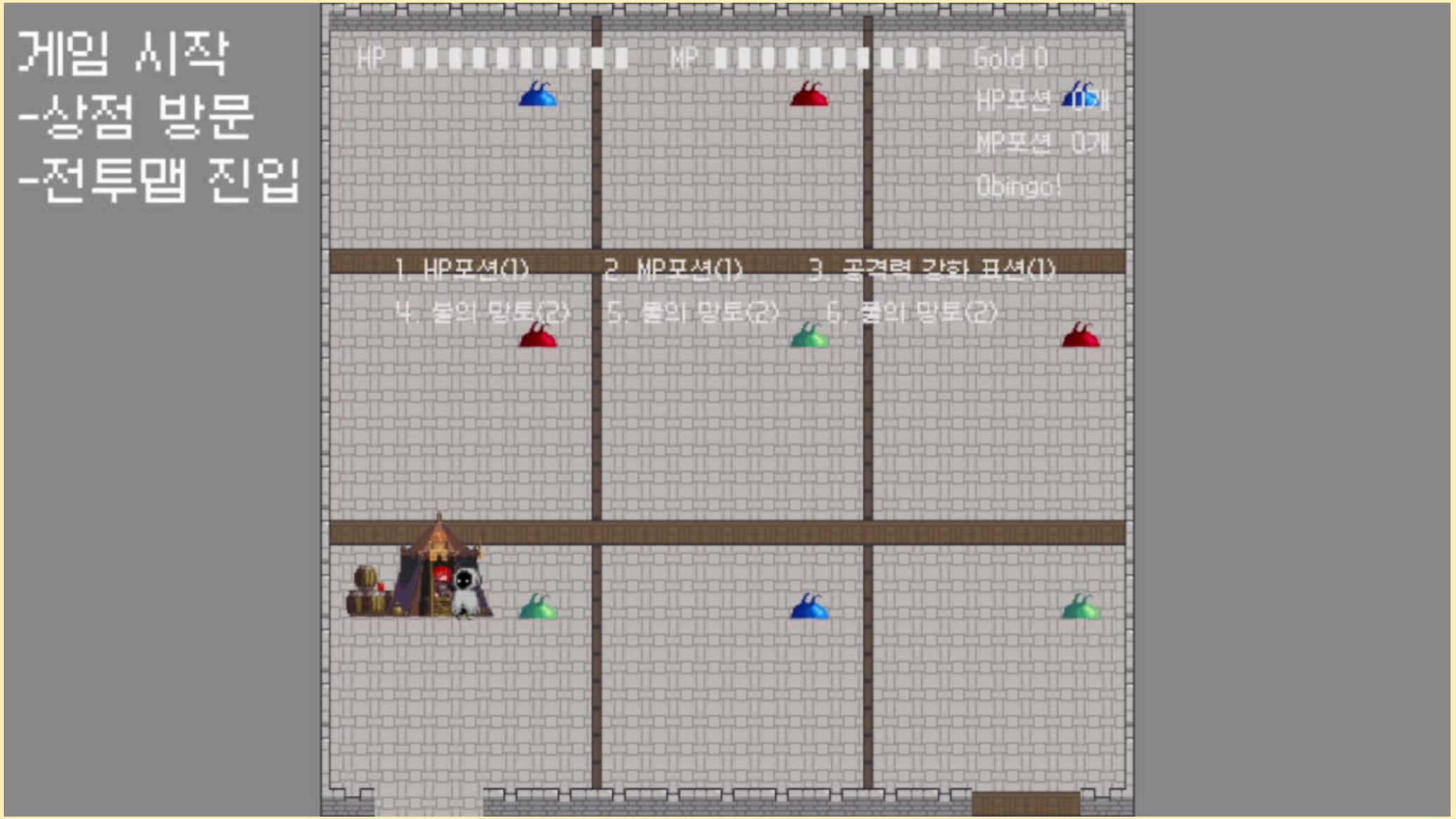
속성 시스템

게임 난이도 조정을 위해 도입한 시스템

종류: 불, 물, 풀
스킬 이름을 통해 자연물로 연상 가능한 상성

속성이 있거나 영향받는 것
-모험가의 스킬
-필드의 슬라임
-상점에서 구매한 망토

프로젝트 소개 - 게임



https://youtu.be/N5p_67uC6Ck

프로젝트 소개 – 게임

https://github.com/seotaerin/BingoRPG.git

class Window

창 생성에 관련된 static 함수들
생성자가 창을 생성할 때 설정된다

Player, Monster, Map, Text 등
다른 클래스에서 이미지&텍스트를
랜더링 할 때 사용된다

```
class Window {
    int frameCount = 0;
    static string bar(int n);
public:
    static SDL_Renderer* renderer;
    static SDL_Window *window;
    static int frame;
    Window();
    static SDL_Texture* LoadTexture(const char* texture);
    void frameCal(int& frame);
    static void UI();
};
```

class Keyboard

currentSituation에 따라
키입력이 다르게 작동하도록 한다

currentSituation
1=빙고맵
2=전투맵
3=상점

```
case SDLK_1: // 2(전투상황)일때는 공격 3(상점)일때는 아이템 구매
    if(currentSituation==2)
    {
        Pointer::renderingFightScene(3);
        if (Pointer::fight->playerattck(0) == 1) Pointer::fightEndCall();
    }
    if(currentSituation==3) Shop::buyItem(1);
    break;
```

프로젝트 소개 – 게임

class Monster

Monster
: 생성자에서 몬스터 이미지설정,
static 변수들 초기화

전투 시작→객체(몬스터) 생성
전투 종료→객체 삭제 (초기화 도움)

```
Monster::Monster()
{
    monsterType=3;
    bossAppear= false;
    gameClear= false;
    monsterFra = { 0 , 0, 64, 64 };
    monsterTexture_l = Window::LoadTexture("slime.png");
    monsterTexture_f = Window::LoadTexture("slime_fire.png");
    monsterTexture_w = Window::LoadTexture("slime_water.png");
    bossTexture = Window::LoadTexture("boss.png");
}
```

class Fight

Fight : 랜덤 체력의 몬스터 생성
UI : 전투맵 내에서의 UI

타입 상성을 포함한 전투식 계산
및 전투상황 출력

```
class Fight {
    int ALLPlayerATK{};
    int monsterhp;
    int temMonterHp;
    int monterATK{};
    int skillType{};
    char* system{};
public:
    static bool fightStart;
    int potionPower;
    Fight();
    void UI();
    int playerattck(int type);
    void monsterattack();
    float playerTypeAttack();
    float monsterTypeAttack();
    void addgold();
    void ItemUse(int n);
```

```
//생성자에서 몬스터체력을 정함
Fight::Fight()
{
    srand(time(nullptr));
    potionPower=0;
    if(Monster::bossAppear) monsterhp = 200;
    else if(Map::bingocount>=2) monsterhp =rand() % 21 + 140;
    else if(Map::bingocount>=1) monsterhp =rand() % 21 + 120;
    else monsterhp = rand() % 21 + 100;
    temMonterHp=monsterhp;
}
```

프로젝트 소개 – 게임

class Pointer

반복 사용되는 다양한 포인터 정의

반복 사용되는
화면 정리 및 출력(렌더) 함수 정의

```
//싸움이 끝날때 파이트 객체 삭제하고 몬스터 죽음, 맵클리어 정보등 불러오는 함수
void Pointer::fightEndCall()
{
    Monster::monsterDie(Player::yPos, Player::xPos);
    player->level1Access[Player::yPos][Player::xPos] = 1;
    fight->addgold();
    player->GameClear();
    if(Monster::bossAppear) Monster::bossAppear= false;

    map->bingoCountCal();
    map->mapClear();
    player->mapClear();

    if(Fight::fightStart)
    {
        delete fight;
        Fight::fightStart= false;
    }
}
```

static 함수
다른 곳에서 사용하기 위한
공유 가능한 변수

```
class Pointer {
public:
    static Window* window;
    static Player* player;
    static Monster* monster;
    static Map* map;
    static Fight* fight;
    static Text* text;
    static void renderingScene();
    static void renderingFightScene(int anim);
    void static fightStartCall();
    void static fightEndCall();
    static void gameOverScene();
    static void gameClearScene();
};
```

동적 할당
게임 초기화시 함께 초기화,
이후 함께 재생성 하기 위함

```
//다른 클래스에서 쓰게 될 포인터들
Window* Pointer::window=new Window();
Player* Pointer::player=new Player();
Monster* Pointer::monster=new Monster;
Map* Pointer::map=new Map();
Fight* Pointer::fight;
Text* Pointer::text=new Text;
```

프로젝트 소개 – 게임

class Player

플레이어의 상태 정보, 인벤토리,
모습, 이동가능 경로 함수 등

class Shop

상점 관리
플레이어가 물품 구매=Player 인벤토리에 물품 추가, 코인 차감

```
int Player::characterX;
int Player::characterY;
int Player::xPos;
int Player::yPos;
int Player::Playerhp;
int Player::Playermp;
int Player::hpPotion;
int Player::gold;
int Player::mpPotion;
int Player::attackPotion;
int Player::playerType;
bool Player::fireCloak;
bool Player::waterCloak;
bool Player::leafCloak;
```

```
//생성자에서 플레이어 초기상태 설정
Player::Player()
{
    characterX = 100;
    characterY = 845;
    xPos=3;
    yPos=27;
    Playerhp=100;
    Playermp=20;
    gold=0;
    hpPotion=0;
    mpPotion=0;
    attackPotion=0;
    playerType=0;
    fireCloak= false;
    waterCloak= false;
    leafCloak= false;
}
```

```
//상점에서 아이템 구매하는 함수 (1 hp포션 2 mp포션 3 공격력 포션)
void Shop::buyItem(int kind)
{
    if(Player::gold>=1)
    {
        Player::gold-=1;

        switch (kind) {
            case 1:
                Player::hpPotion+=1;
                break;
            case 2:
                Player::mpPotion+=1;
                break;
            case 3:
                Player::attackPotion+=1;
                break;
            default:
                break;
        }
```

static 상수
상점 품목과 동명의
case = 가독성

프로젝트 소개 - 게임

초안

고려사항
-에셋 가격
-스프라이트 종류와 양
-적절한 해상도

발생한 문제
-다양한 오류 (초기화 등)
-라이브러리 사용
-콘솔&그래픽 동시사용

완성안

-층계 구현
-층별 몬스터 차등 (난이도 조절)
-상점 스킬 판매

+보스 구현
+속성 시스템 도입 (난이도 조절)
+상점 속성 방어 아이템 판매

개발 일정


9주차
10주차
11주차
12주차
13주차
14주차
기획 수정
에셋 서치, 기획 수정
전투 시스템 확립, 레벨 디자인
맵 작성
3x3 나누기
몬스터 배치
방 클리어 구현
전투 작성
전투 스테이터스 구현
전투 구현
코드 추합
맵 & 전투 연결, UI
디버깅
발표 준비

Q & A

THANKS