



어소트락 아카데미

# Blasphemous Imitation DirectX11

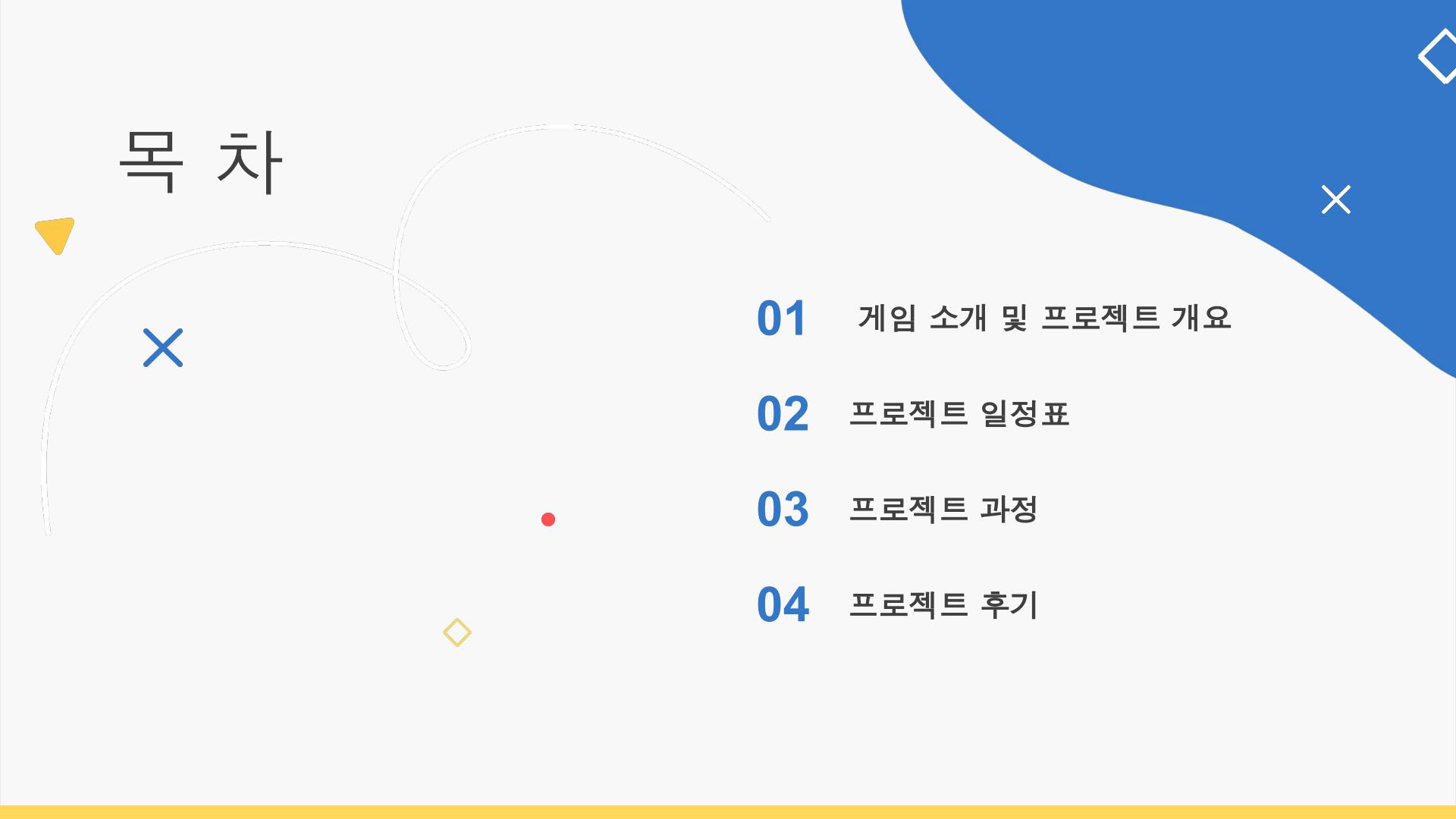


KDT 4기 – 전유성



고용노동부

# 목 차

- 
- 01** 게임 소개 및 프로젝트 개요
  - 02** 프로젝트 일정표
  - 03** 프로젝트 과정
  - 04** 프로젝트 후기

# 01 게임 소개 및 프로젝트 개요

1

## 게임 소개 및 개요

블라스퍼머스 게임은 유니티 엔진을 기반으로 제작된 2D 메트로배니아 다크 판타지 게임으로, 재미있게 클리어한 경험도 있었고, 직접 만들어보고 싶어서 선택

2

## 프로젝트 목표

보스전으로 컨텐츠를 구상하였고, 최종 보스를 1페이지, 2페이지로 나눠서 구현

3

## 활용한 도구

Visual Studio,  
DirectX11

4

## 프로젝트 구조

DirectX11 기반으로  
포트폴리오 제작

5

## 기대 효과

자체 엔진을 제작으로  
통한 그래픽스 API 및  
엔진 이해를 하여 게임  
제작에 대한 전반적인  
과정을 학습

## 02 프로젝트 일정표

구분	기간	활동		비고
사전 기획	2025 11/30 ~ 2024 12/02	🚩 프로젝트 기획 및 사전 작업	🚩 리소스 수집, 개발 계획 수립	
단계별 개발(1/4)	2025 12/03 ~ 2025 12/10	🚩 이미지 편집 및 정리	🚩 게임 구조 설계	
단계별 개발(2/4)	2025 12/11 ~ 2025 12/17	🚩 게임 구조 설계 마무리	🚩 캐릭터 기본 구조 구현	
단계별 개발(3/4)	2025 12/18 ~ 2025 12/24	🚩 캐릭터 스킬 및 회복 기능 구현	🚩 보스 1페이즈 구현	
단계별 개발(4/4)	2025 12/25 ~ 2025 12/31	🚩 보스 2페이즈 구현 및 보스 스킬 패턴 추가		
추가 작업 및 디버깅	2025 01/01 ~ 2026 01/05	🚩 디테일이나 추가 작업	🚩 에러 확인 및 디버깅	
총 개발기간	2025 11/30 ~ 2026 01/05			

# 03 프로젝트 과정 – 형상 관리

## ▶ Tortoise SVN 형상관리

SVN  
형상  
관리

The screenshot shows the Tortoise SVN client interface. At the top, there's a search bar and date filters set from 2025-12-01 to 2026-01-05. Below is a table of commit history:

Revision	Actions	Author	Date	Message
105	🕒 📄	JYS	2025년 12월 9일 화요일 오후 5:53:27	1208 2026 플레이어 - 조건을 줘서 움직임이 0이면 움직임을 멈추는 기능 추가 20...
104	🕒 📄	JYS	2025년 12월 8일 월요일 오후 6:41:51	플레이어기능 12.08 버그 많음 현재 완성도 3% 1. 이미지 이슈 2. 회복 및 죽음은 엠...
103	🕒 📄	JYS	2025년 12월 8일 월요일 오후 6:29:24	회복 및 죽음 기능은 엘버변수로 처리할 예정
102	🕒 📄	JYS	2025년 12월 8일 월요일 오전 2:06:15	
101	🕒 📄 ✎	JYS	2025년 12월 8일 월요일 오전 2:02:05	
100	🕒 📄	JYS	2025년 12월 5일 금요일 오후 5:49:26	
99	🕒 📄	JYS	2025년 12월 4일 목요일 오후 6:22:30	목요일 작업 달.orm
98	🕒 📄 ✎	JYS	2025년 12월 4일 목요일 오전 9:31:02	
97	🕒 📄	JYS	2025년 12월 4일 목요일 오전 1:29:39	질문 1204 - 씬메인, 플레이어 오브젝트, 플레이어 무브 구조 비슷할거니 하나만 ...
96	🕒 📄	JYS	2025년 12월 3일 수요일 오후 6:42:43	1203 플레이어 기본 구조만 잡음 클래스 상속 받아서 해결하기 이동, 점프, 공격, ...
95	🕒 📄	JYS	2025년 12월 3일 수요일 오전 12:56:...	디바이스에서 폰트 초기화가 안되어서 문제가 발생함 -> 수정 콘텐츠 짜기 계속하...
94	🕒 📄	JYS	2025년 12월 2일 화요일 오후 5:49:12	1202 사진 정리 및 폰트 오류 수정하기 플레이어 이미지 더 봄아보기

Below the commit history, there are two sections of file changes:

Path	Action	Copy from path	Revision
↳ /BlasStart1201/Blasphemous/include/Object/BulletObject.h	Modified		
*↳ /BlasStart1201/Blasphemous/include/Object/GunnerMonster.cpp	Modified		
*↳ /BlasStart1201/Blasphemous/include/Object/PlayerObject.cpp	Modified		
↳ /BlasStart1201/Blasphemous/include/Object/PlayerObject.h	Modified		
*↳ /BlasStart1201/Blasphemous/include/Scene/SceneAssetManager.cpp	Modified		
↳ /BlasStart1201/Blasphemous/include/Scene/SceneAssetManager.h	Modified		
*↳ /BlasStart1201/Blasphemous/include/Scene/SceneMain.cpp	Modified		
*↳ /BlasStart1201/Blasphemous/include/UI/Common/Button.cpp	Modified		
*↳ /BlasStart1201/Blasphemous/include/UI/UserWidget/MainWidget.cpp	Modified		
*↳ /BlasStart1201/Blasphemous/include/UI/Widget.cpp	Modified		
↳ /BlasStart1201/Blasphemous/include/UI/Widget.h	Modified		

At the bottom, there's a summary message: "SVN을 이용하여 프로젝트의 버전 관리".

SVN을 이용하여 프로젝트의 버전 관리

# 03 프로젝트 과정 - 프로젝트 폴더 관리

## ▶ 폴더 경로 정리

### 폴더 관리

폴더 경로			
출력 디렉터리	./Bin/		
중간 디렉터리	./BinObj/		
대상 이름	\$(ProjectName)_Debug		
구성 형식	애플리케이션(.exe)		
Windows SDK 버전	Windows 10.0 SDK(최신)		
플랫폼 도구 집합	v145		
Animation		2025-12-23 오후 4:39	파일 폴더
Asset		2025-12-02 오후 5:22	파일 폴더
Component		2026-01-05 오전 9:45	파일 폴더
Object		2026-01-06 오후 2:23	파일 폴더
Render		2026-01-05 오전 10:18	파일 폴더
Scene		2026-01-05 오후 12:48	파일 폴더
Shader		2025-12-10 오후 6:28	파일 폴더
Share		2025-12-31 오전 9:19	파일 폴더
UI		2025-12-29 오전 8:38	파일 폴더

Visual Studio에서 출력 디렉터리 및 중간 디렉터리의 경로 지정하여 폴더 관리 및 파일 확장에 용이하도록 설정.

# 03 프로젝트 과정

- ▶ 실제 게임과 비슷하게 구현하는게 목표

## 게임 비교



실제 게임

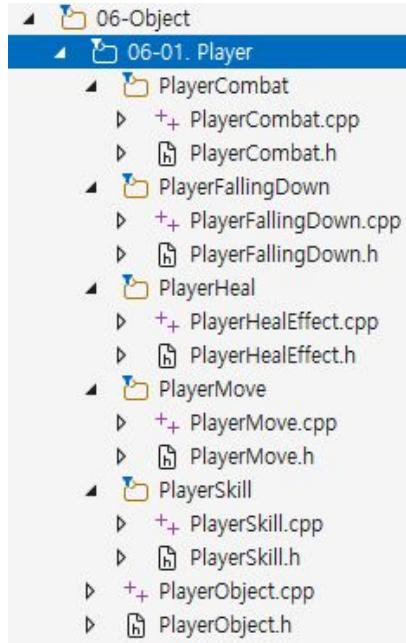


모작 게임

# 03 프로젝트 과정 – 캐릭터

## ▶ 캐릭터 기본 구조 구현

캐릭터



PlayerObject 파일 기반으로, 전투, 낙하, 회복, 이동, 스킬 로직을 구현

PlayerObject와 상호작용하는 파일들은  
Is-a 관계가 아닌 Has-a 관계로 설정하여 구현

## 03 프로젝트 과정 – 캐릭터

## ▶ 캐릭터 기본 구조 구현

## 캐릭터

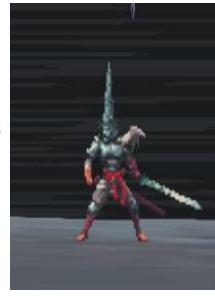
```
private:
    float mGravity = -3000.f; // 중력
    float mVelocityY = 10.f; // Y축 속도
    float mTerminalVelocity = -4000.f; // 중력 속도

void CGravityComponent::Update(float deltaTime)
{
    if (!mOwnerObject) return;

    // 공중이면 중력 적용
    if (!mGrounded)
    {
        mVelocityY += mGravity * deltaTime;

        // 속도(낙하·최대·속도) 제한
        if (mVelocityY < mTerminalVelocity)
            mVelocityY = mTerminalVelocity;

        // 월드·포지션·직접·적용하여·Y축·이동.
        mOwnerObject->AddWorldPos(0.f, mVelocityY * deltaTime, 0.f);
    }
    else
    {
        // 착지 상태에서는 중력이 당기는 힘을 제거
        if (mVelocityY < 0.f)
            mVelocityY = 0.f;
    }
}
```



```
    ....const float velY=gravity->GetVelocityY();
....if(velY>-0.f)
....{
....    gravity->SetGrounded(false);
....    return;
....}

....const float playerMinY=playerCollider->GetMin().y;
....const float floorMaxY=floorCollider->GetMax().y;

....const float penetration=floorMaxY-playerMinY;

....if(penetration>-0.5f)
....{
....    if(penetration>0.f)
....    {
....        FVector3D pos=Obj->GetWorldPosition();
....        pos.y+=penetration;
....        Obj->SetWorldPos(pos);
....    }
....}

....gravity->SetGrounded(true);
```



콜라이더를 이용하여 바닥 충돌 체크 및 중력 구현으로 플레이어가 점프할 수 있도록 구현

# 03 프로젝트 과정 – 캐릭터

## ▶ 캐릭터 전투 로직 구현

Combat

```
void CPlayerCombat::StartCombo(int idx)
{
    > ResetHit();
    > mComboIndex+=idx;
    |
    > if(idx==1){.
    >     mAnimation->ChangeAnimation("PlayerAttack1");
    > }
    > else if(idx==2){.
    >     mAnimation->ChangeAnimation("PlayerAttack2");
    > }
    > else{.
    >     mAnimation->ChangeAnimation("PlayerAttack3");
    > }

    > if(mATKCollider)mATKCollider->SetEnable(false);
    > if(mUpATKCollider)mUpATKCollider->SetEnable(false);
}

void CPlayerCombat::OnParryInput(bool bFlip)
{
    > if(!mAnimation) return;
    > if(mAtkType!=EPlayerAtkType::None) return; // 공격 중 패링 금지

    > mParryActive+=true;
    > mParrySuccess+=false;

    > mAnimation->ChangeAnimation("PlayerParry");

    > if(!mParryCollider) return;

    // 패링 창 위치
    > if(bFlip)mParryCollider->SetRelativePos(-40.f,-0.f);
    > else.....mParryCollider->SetRelativePos(40.f,-0.f);

    > mParryCollider->SetEnable(true);
}
```



공격은 z를 누르면 기본 공격을 하며, 몬스터 피격 성공시, 다음 콤보로 넘어갈 수 있게 만듬

패링 기능은 x키를 누르면 발동되며, 패링 성공시 데미지를 입지 않게 구현

# 03 프로젝트 과정 – 캐릭터

## ▶ 캐릭터 전투 로직 구현

### Combat

```
void CPlayerCombat::StartUpAtk()
{
    → ResetHit();
    → mAnimation->ChangeAnimation("PlayerUpATK");
    → if (mATKCollider) · mATKCollider->SetEnable(false);
    → if (mUpATKCollider) · mUpATKCollider->SetEnable(false);
}

void CPlayerCombat::StartDodgeAtk()
{
    → ResetHit();
    → mAnimation->ChangeAnimation("PlayerDodgeATK");

    → mDodgeAtkDash -= true;
    → mDodgeAtkDashAcc -= 0.f;

    → if (mATKCollider) · mATKCollider->SetEnable(false);
    → if (mUpATKCollider) · mUpATKCollider->SetEnable(false);
}
```



A키를 누르면 위로 때리는 공격  
애니메이션이 실행, 콜라이더 위치에  
몬스터가 히트시 데미지를 입음.

S키를 누르면 회피 공격이 발동,  
움직이면서 공격을 함

# 03 프로젝트 과정 – 캐릭터

## ▶ 캐릭터 스킬 구현

### Skill

```
//스킬.바인드()
void SkillBind(CPlayerObject* owner, CAnimation2D* playerAnim,
               const CSharedPtr<CSpriteComponent>& beamSprite, CAnimation2D* beamAnim,
               CColliderAABB2D* beamCollider){
    mOwner = owner;
    mPlayerAnim = playerAnim;
    mBeamSprite = beamSprite;
    mBeamAnim = beamAnim;
    mBeamCollider = beamCollider;
}

//스킬..콜라이더.셋팅()
mSkillBeamSprite->AddChild(mSkillCollider);
mSkillCollider->SetBoxSize(350.f, 1000.f);
mSkillCollider->SetRelativePos(0.f, 0.f, 0.f);
mSkillCollider->SetCollisionProfile("PlayerAttack");
mSkillCollider->SetEnabled(false);
mSkill->skillBind(this, mAnimation, mSkillBeamSprite, mSkillBeamAnim, mSkillCollider.Get());
mSkill->Setup();
```

Skill Class로 나눠 스킬 이펙트 On/off하게 구현  
SkillBind 함수를 통해서 PlayerObject에서  
Setting한 걸 바인드



# 03 프로젝트 과정 – 캐릭터

## ▶ 캐릭터 회복 구현

Heal

```
bool CPlayerHealEffect::Setup()
{
    if (!mFXSprite || !mFXAnim)
        return false;

    if (mReady)
        return true;

    if (!mFXAnim->AddSequence("HealFX", -3.0f, 1.2f, true, false))
    {
        CLog::PrintLog("HealFX>AddSequence FAILED");
        mReady = false;
        return false;
    }

    mFXAnim->SetEndFunction("HealFX", this, &CPlayerHealEffect::HealEnd);
    mFXSprite->SetEnable(false);
    mReady = true;
    return true;
}
```

▶ PlayerObject.cpp

PlayerObject.h

캐릭터가 회복 모션을 할 때, 그 위에 이펙트가 발동하도록 구현

```
void CPlayerHealEffect::Play(bool bFlip)
{
    if (!mFXSprite || !mFXAnim)
        return;

    if (!mReady && !Setup())
        return;

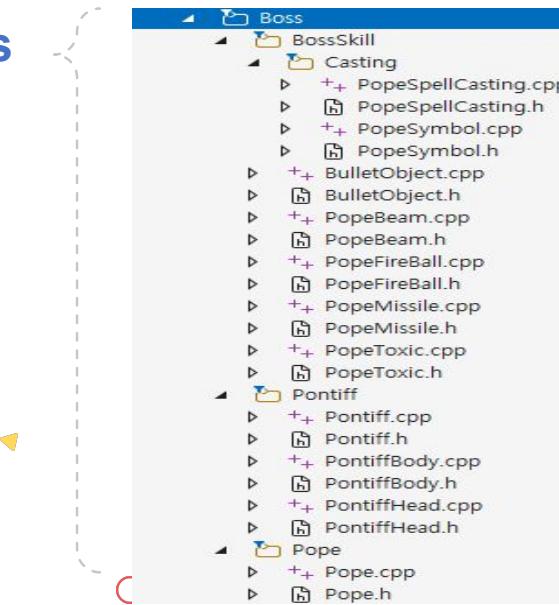
    mFXSprite->SetFlip(bFlip);
    mFXAnim->ChangeAnimation("HealFX");
    mFXSprite->SetEnable(true);
}
```



# 03 프로젝트 과정 – 보스

## ▶ 보스 구현

Boss



보스

Pope, Pontiff 기반으로, 페이즈 1과 페이즈 2로 나눴으며, 다양한 스킬을 구현

스킬 대부분은 투사체로 BulletObject 기반으로 상속하여 FireBall, Missile, Toxic 구현

# 03 프로젝트 과정 – 보스 페이즈1

## ▶ 보스 페이즈 1 구현

Boss

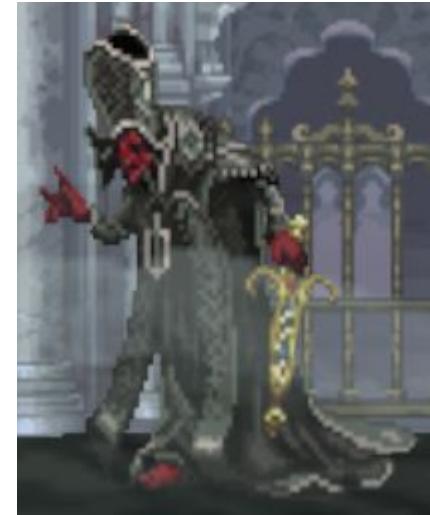


```
// 교황의 시점은 항상 플레이어에게 향하게
if(mTarget)
{
    // 죽었거나 비활성/비표시 상태면 타겟 해제
    if(!mTarget->IsActive())
    {
        mTarget=nullptr;
    }
    else if(mTarget->IsEnable())
    {
        const float popeX=mTarget->GetWorldPosition().x;
        const float targetX=mTarget->GetWorldPosition().x;

        // 타겟이 왼쪽이면 flip=true;
        mRoot->SetFlip(targetX<popeX);
    }
}

void CPope::DecideTeleportThreshold()
{
    // 스클립은 2번 아니면 3번
    mTeleportThreshold=2+(rand()%2); // 2 or 3
}

void CPope::OnVanishingEnd()
{
    TeleportToEdgeRandom();
    // 순간이동 후에도 타겟 잡기
    if(mTarget&&mTarget->IsEnable()){
        mRoot->SetFlip(mTarget->GetWorldPosition().x<GetWorldPosition().x);
    }
    // 이동 후 등장
    mPopeDefault=EPopState::PopeAppear;
}
```

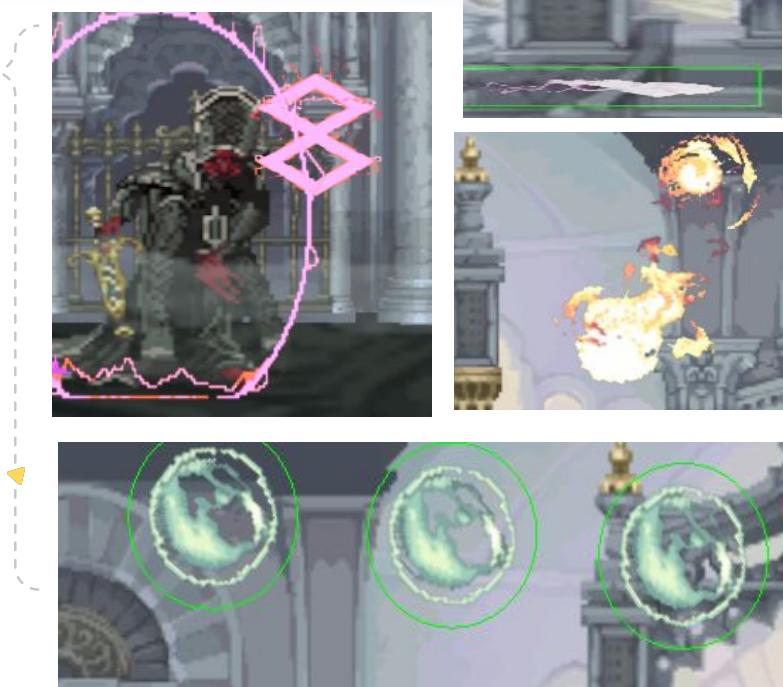


항상 플레이어를 바라보게 구현  
스킬 2번 or 3번을 사용하고나서 순간이동하여  
특정 좌표로 이동

# 03 프로젝트 과정 – 보스 페이즈1

## ▶ 보스 스킬 구현

### Boss Skills



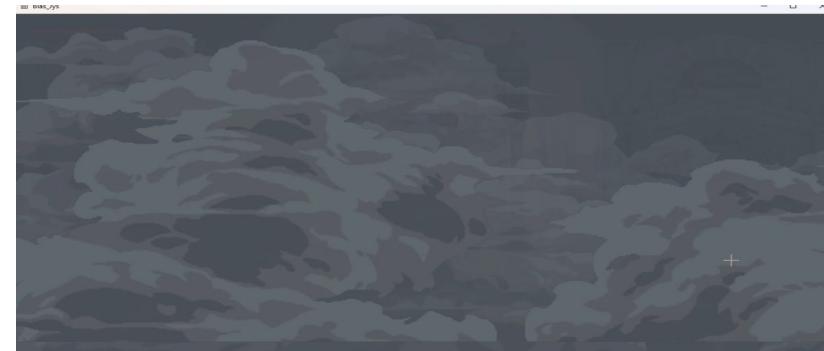
보스가 스킬 시전시 이펙트가 발생되며 Fireball, Toxic, Missile의 패턴에 따라 보스의 이펙트 색상이 달라짐

Fireball, Toxic은 플레이어로 향하게 타겟 설정

# 03 프로젝트 과정 (보스 – 씬 전환)

## ▶ 씬 전환

### Scene



보스 페이즈1이 죽으면 안개가 등장하며  
페이즈2가 등장

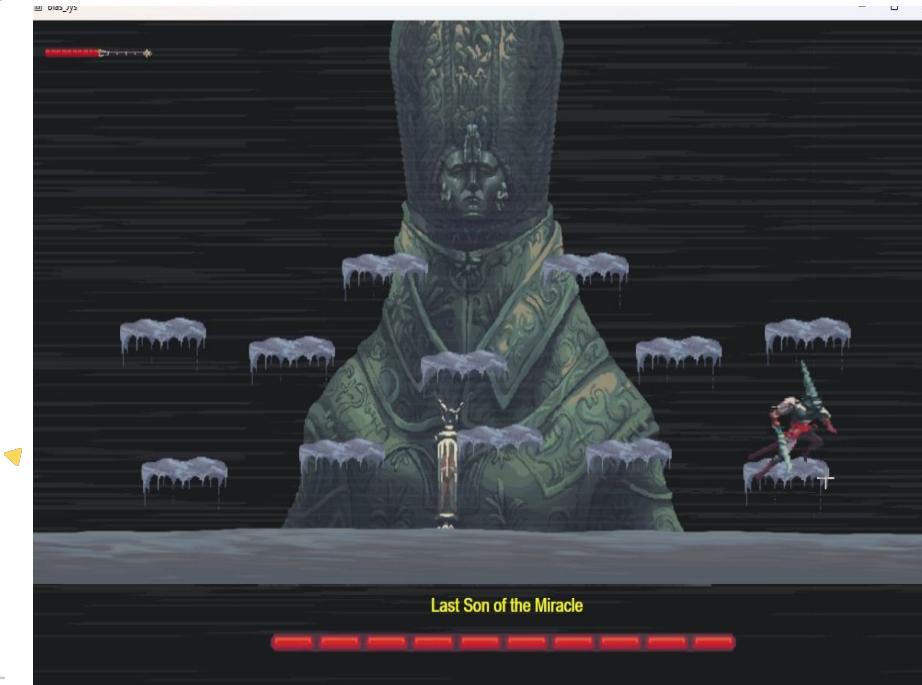
페이즈2로 씬 전환하기 전에 위젯으로  
안개가 뜨면서 전환

# 03 프로젝트 과정 - 보스 페이즈2

## ▶ 보스 페이즈2 구현

Boss

x

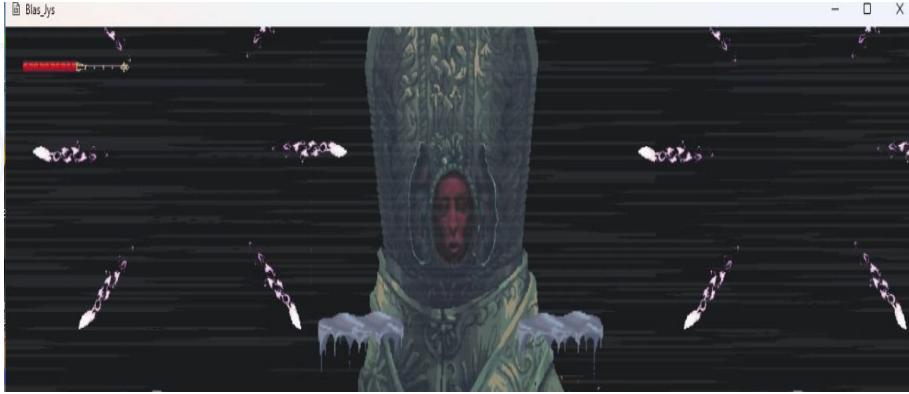


보스 페이즈2

얼굴이 열릴 때만 때릴 수 있음

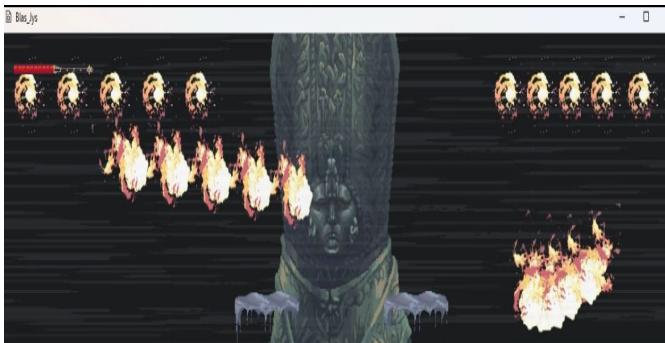
발판은 특정 시간마다 생성되었다가  
사라졌다가를 반복함

# 03 프로젝트 수행 경과 (보스 – 페이즈 2)



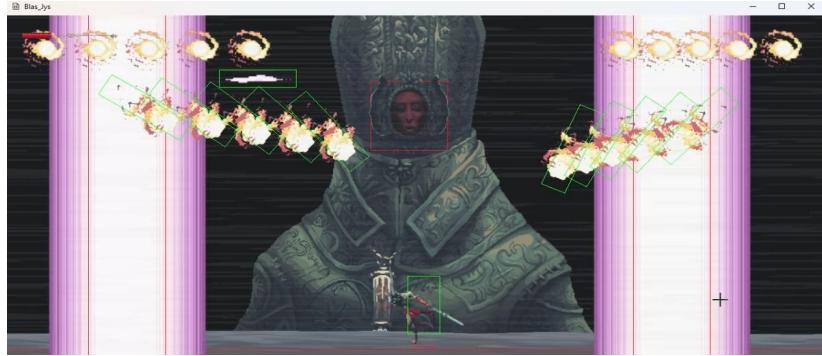
보스 페이즈 1처럼 Fireball, Toxic, Missile 사용

페이즈1와 다르게 패턴 변환을 줌



# 03 프로젝트 과정 - 보스 페이즈2

## ▶ 보스 페이즈2 구현



스킬 패턴 중에 빔 스킬을 추가

빔 스킬은 끝에서 2발 발사 한뒤에 5초  
딜레이 후 안쪽에서 2발 발사



# 03 프로젝트 과정 – 기타 UI 작업

## ▶ 기타 UI 작업



타이틀 화면 제작

Start, Edit 버튼 존재

Start 클릭시 보스 페이즈1로 진입

Edit 버튼 클릭시 타일맵 에디터로 전환

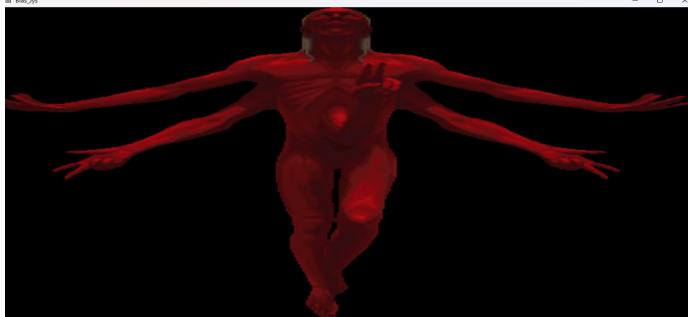


플레이어가 죽을 시

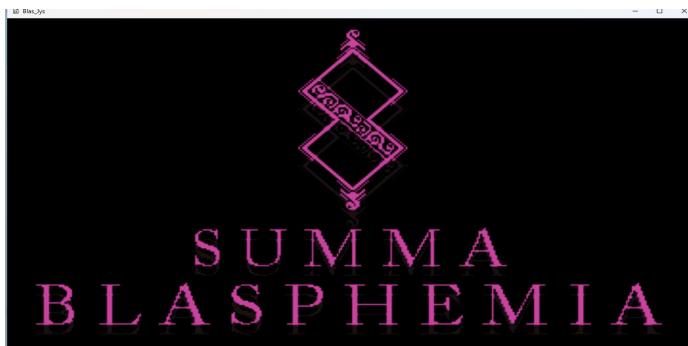
Excommunicationis 위젯 화면이 뜨며 게임 오버

# 03 프로젝트 과정 – 기타 UI 작업

## ▶ 기타 UI 작업



보스 페이즈2가 죽을 때 발동 되는 위젯



첫 번째 위젯이 나온뒤에 3초뒤에 두 번째  
위젯인 Summa Blasphemia가 나오며 게임은  
끝남

# 04 프로젝트 후기

## ▶ 프로젝트 마친 소감

### 프로젝트 결과물에 대한 완성도 평가

평점 5/10

초기 계획보다 계속 수정하여 원하는 퀄리티가 안 나옴, 초안을 생각보다 크게 작성해서 일정 관리 실패 함

### 프로젝트 결과물의

### 추후 개선점이나 보완할 점 등 내용 정리

- UI 추가
- 애니메이션 크기가 다 달라서 크기 맞춰야 함
- 애니메이션이 빨리 움직이는 경우가 있는데, 플레이 시간을 제대로 잡아줘야 함.

### 프로젝트 후기

첫 프로젝트 작품이라 걱정이 컸는데, 무사히 결과물이 나왔음. 다만, 일정 관리가 잘 안 되었고, 생각보다 원하는 퀄리티가 아니라서 로직을 다시 만드는 등, 아쉬운점이 많은 프로젝트라고 생각함

### 프로젝트를 수행하면서 느낀 점

- 프로그래밍 작업을 하면서 부족한 개념을 복습
- 혼자 만든 작품이지만 비슷한 게임을 만드는 친구들과 의사소통 통해 더 좋은 아이디어를 얻음
- 일정 관리가 매우 중요하다는 걸 깨달음

# 05 프로젝트 – 시연 영상

