(회사이름)

클라이언트(지원하는 직책)

PORTFOLIO

PORTFOLIO

CONTENTS

1. 소개

2. 프로젝트

   1) Unity
   2) Unreal
   3) DirectX 12
   4) 기타

1. 소개

홍 진 선

I AM

- 출생 : 1996.09.13
- 전화번호 : 010 - 2052 - 2419
- E-mail : hjs0913@naver.com
- 학력 : 한국공학대학교 게임공학과 졸업예정

SKILLS

개발 도구

엔진

- Unity
- Unreal

Visual Studio

- DirectX 12
- OpenGL

언어

- C, C++
- C#, Python

기타

- MS SQL
- Github

서버

- IOCP
- 소켓 프로그래밍

2. 프로젝트

UNITY

현장실습

- 기간 : 21.07 ~ 21.08
- 내용 : 하이퍼 캐쥬얼 게임 제작

DIRECTX 12

팀 프로젝트

- 기간 : 22.01 ~ 22.07
- 내용 : 3인칭 MMORPG 게임 제작

기타

팀 프로젝트

- 기간 : 21.05 ~ 21.06
- 내용 : Python을 활용한 어플리케이션 개발

UNREAL

개인 프로젝트

- 기간 : 21.05 ~ 21.06
- 내용 : 3인칭 FPS 게임 제작

기타

팀 프로젝트

- 기간 : 21.10 ~ 21.12
- 내용 : 소켓 프로그래밍을 활용한 네트워크 게임 제작

1) UNITY (현장실습)

(영상 첨부)

1) UNITY (현장실습)

```
    참조 9개
    public static class Prefab
    {
        public static readonly string Root = "Prefabs/UIRoot";
        public static readonly string Background = "Prefabs/Background";
        public static readonly string Map = "Prefabs/Map";
        public static readonly string Title = "Prefabs/TitleUI";
        public static readonly string Dead = "Prefabs/DeadUI";
        public static readonly string Clear = "Prefabs/ClearUI";
        public static readonly string Button = "Prefabs/ButtonUI";
        public static string Player = "Prefabs/Player";
        public static string Enermy = "Prefabs/Enermy";
    }
}

참조 1개
public static class Bundle
{
    참조 1개
    public static T LoadAsset<T>(string path) where T : Object
    {
        return Resources.Load<T>(path);
    }
}

참조 9개
public static class Util
{
    참조 9개
    public static GameObject InstantiatePrefab(string path, Transform parent)
    {
        return GameObject.Instantiate(Bundle.LoadAsset<GameObject>(path), parent);
    }
}
```

- 에디터를 사용하지 않고 C# 스크립트를 사용하여 게임 제작이 1차 목표였기 때문에 프리팹을 적극 활용하였다.

- 해당 코드는 만든 프리팹을 리소스로 불러온 후 인스턴스화 하여 게임오브젝트로 만들었다.

- 재사용 가능한 클래스로 약간의 수정으로 어디서든 활용할 수 있다.

1) UNITY (현장실습)

```
public void Update()
{
    m_camera.Update();
    if (m_UIManager.isStart == true)
    {
        Debug.Log("재시작!");
        PlayGame();
        m_UIManager.isStart = false;
    }

    if (m_colliderManager.isDead == true)
    {
        Debug.Log("악! 내가 죽었뜨아!");
        m_UIManager.deadUI.SetActive(true);
        Time.timeScale = 0;
        m_colliderManager.isDead = false;
    }

    if (m_playerController.GetScore() == m_enermyGenerator.GetMaxEnermy())
    {
        Debug.Log("모든 적 처지!!");
        Time.timeScale = 0;
        m_UIManager.clearUI.SetActive(true);
    }

    m_deltaTime = Time.deltaTime;

    m_playerController.Update();
    m_enermyGenerator.Update(m_deltaTime);
    m_colliderManager.Update();
    m_UIManager.SetScoreText(m_playerController.GetScore());
}
```

- 해당 코드는 Monobehaviour의 Update함수가 매 프레임마다 호출 되고 변경사항이 없어도 조건문을 검사하기 때문에 비효율적이다.

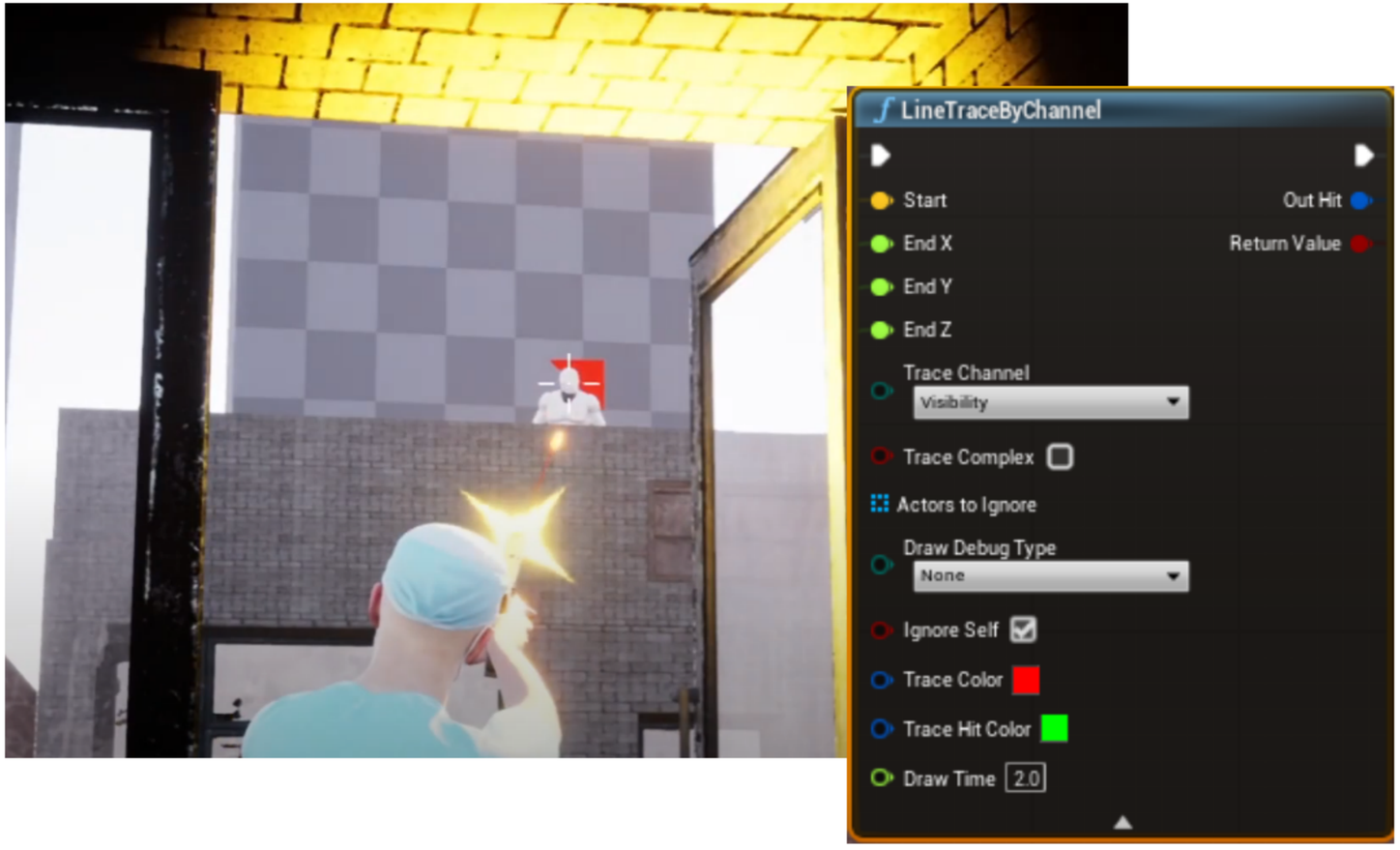
- 이를 해결하기 위해 이벤트 트리거를 생성하고 변경사항이 생기면 이벤트를 호출하여 조건문을 검사하는 방식을 사용하기 위해 이벤트 트리거를 학습하였다.

- 추가적으로 개발할 때 마다 복잡해지는 코드를 간결하게 만들고, 수정/보완을 쉽게 하기 위해 디자인 패턴을 공부하였다.

2) UNREAL (개인 프로젝트)

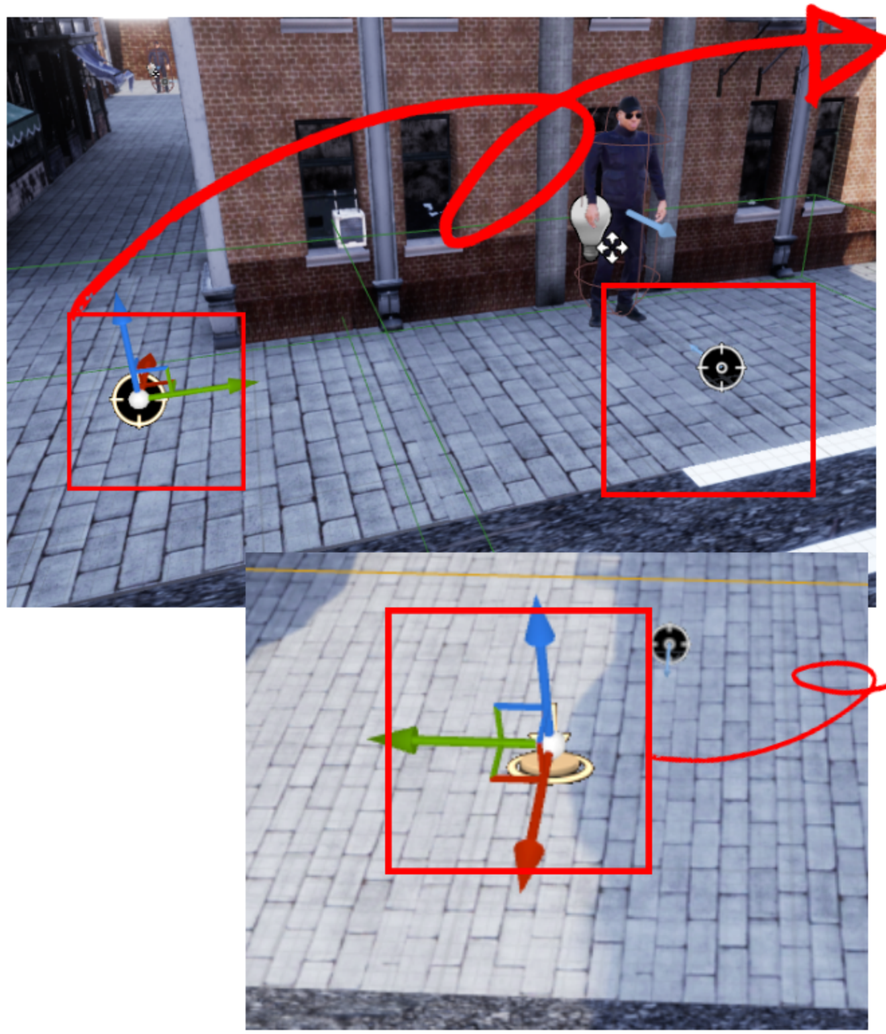
(영상 첨부)

2) UNREAL (개인 프로젝트)



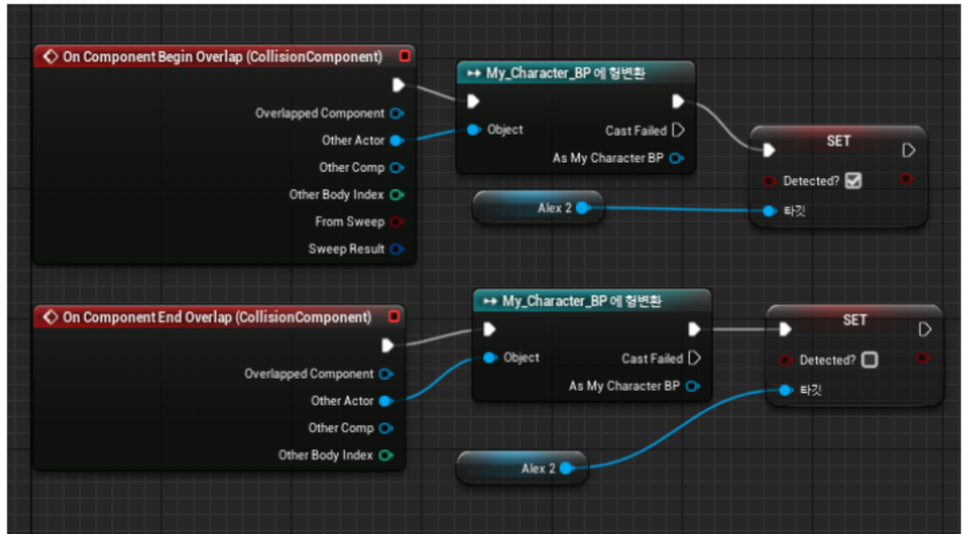
- LineTraceByChannel로 피격 판정을 하고, 피격 받은 위치의 좌표를 받아와서 총에 맞은 것 처럼 피가 나게 만들었다.

2) UNREAL (개인 프로젝트)



- TargetPoint를 사용하여 AI 적의 순찰 범위를 지정해 주었다.

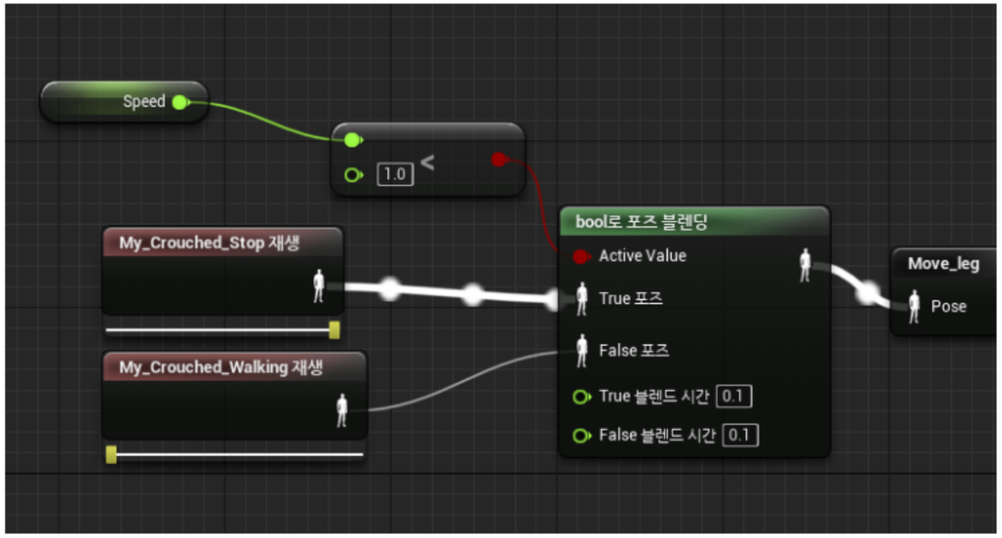
- TriggerBox로 플레이어가 Overlap 되는지 확인 후 순찰 하는 AI가 플레이어를 공격할지 말지를 결정한다.



2) UNREAL (개인 프로젝트)

- 모델의 스켈레톤 트리에 총을 장착할 GunSocket을 생성하고 권총의 프리뷰 에셋을 추가했다.

- 블렌딩을 통해 걸을 때와 멈췄을 때의 애니메이션을 다르게 재생한다.



(GIF 첨부)

- 걸을 때와 멈췄을 때, 조준할 때의 에임 크기가 변화한다.

3) DIRECTX 12 (팀 프로젝트)

(영상 첨부)

3) DIRECTX 12 (팀 프로젝트)

```
XMFLOAT3 xmf3ObjectPos = XMFLOAT3(mPlayer[vec]->GetPosition().x, mPlayer[vec]->GetPosition().y + raidY, mPlayer[vec]->GetPosition().z);
XMFLOAT3 xmf3ViewProj = Vector3::TransformCoord(Vector3::TransformCoord(xmf3ObjectPos, camera->GetViewMatrix()), camera->GetProjectionMatrix());

float fScreenX = xmf3ViewProj.x * (FRAME_BUFFER_WIDTH / 2) + FRAME_BUFFER_WIDTH / 2;
float fScreenY = -xmf3ViewProj.y * (FRAME_BUFFER_HEIGHT / 2) + FRAME_BUFFER_HEIGHT / 2;
```

- xmf3ObjectPos : 오브젝트의 월드 좌표
- xmf3ViewProj : 오브젝트의 월드 좌표를 뷰변환, 투영변환 한 좌표
- fScreenX,Y : 변환된 좌표를 화면상 좌표에 맞춰 주기 위한 변수

3) DIRECTX 12 (팀 프로젝트)

```
CTexture* pTerrainRoadTexture = new CTexture(1, RESOURCE_TEXTURE2D, 0);
if (!InDungeon) {
    pTerrainRoadTexture->LoadTextureFromFile(pd3dDevice, pd3dCommandList, L"Terrain/stone_ground.dds", 0);
}
```

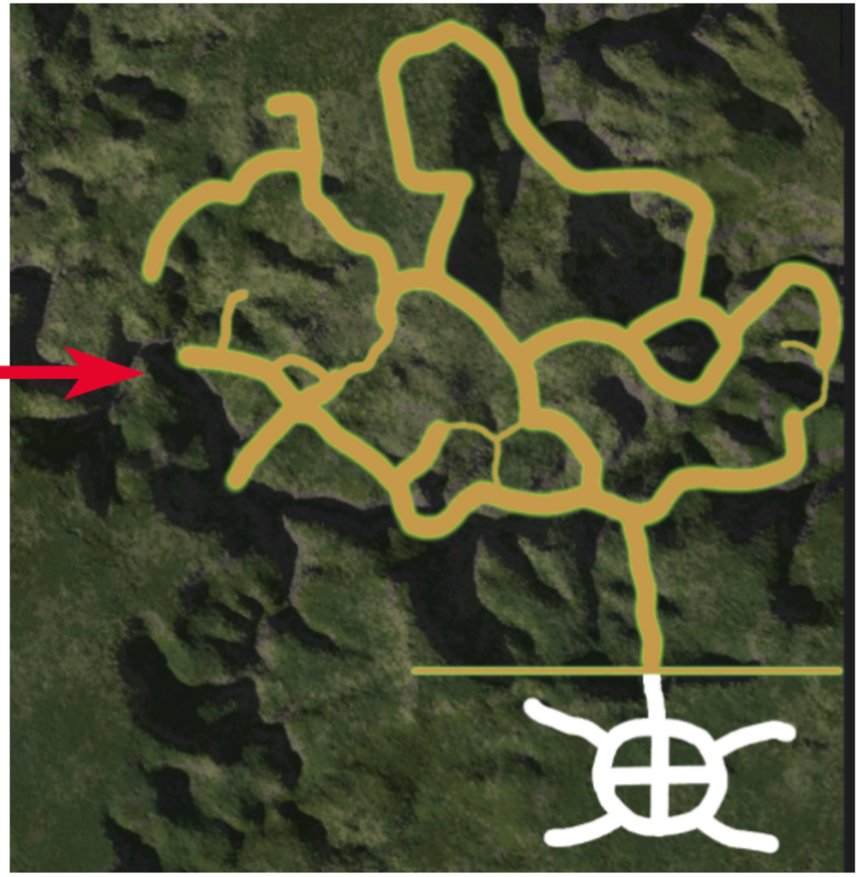
(Scene.cpp)

```
    float4 cBaseTexColor = gtxtTerrainBaseTexture.Sample(gssWrap, input.uv0);
    float4 cDetailTexColor = gtxtTerrainDetailTexture.Sample(gssWrap, input.uv1);
    float4 cRoadTexColor = gtxtTerrainRoadTexture.Sample(gssWrap, input.uv2);
float4 field = float4(0.772f, 0.604f, 0.289f, 1.0f);
float4 city = float4(1.f, 1.0f, 1.0f, 1.0f);

if (cBaseTexColor.x == city.x && cBaseTexColor.y == city.y && cBaseTexColor.z == city.z)
    cColor = saturate((cBaseTexColor * 0.2f) + (cRoadTexColor * 1.0f));

if (cBaseTexColor.x >= (field.x - 0.005f) && cBaseTexColor.x <= (field.x + 0.005f)
    && cBaseTexColor.y >= (field.y - 0.005f) && cBaseTexColor.y <= (field.y + 0.005f)
    && cBaseTexColor.z >= (field.z - 0.005f) && cBaseTexColor.z <= (field.z + 0.005f))
    cColor = input.color * saturate((cBaseTexColor * 0.5f) + (cRoadTexColor * 1.0f));
```

(Shaders.hlsl)

- BaseTexture의 색깔에 따라 RoadTexture를 렌더링 할지 말지를 정하는 코드

- 길의 모양이 구불구불해서 특정 위치만 타일링 하기 힘들다고 생각해서 길 자체에 색깔을 지정해주고 색깔정보를 판단하여 다른 텍스쳐를 렌더링하게 만들었다.



(BaseTexture)