

언리얼 CPP

1강

목차

1. 개요
2. cpp와 블루프린트 비교
3. 실습

개요

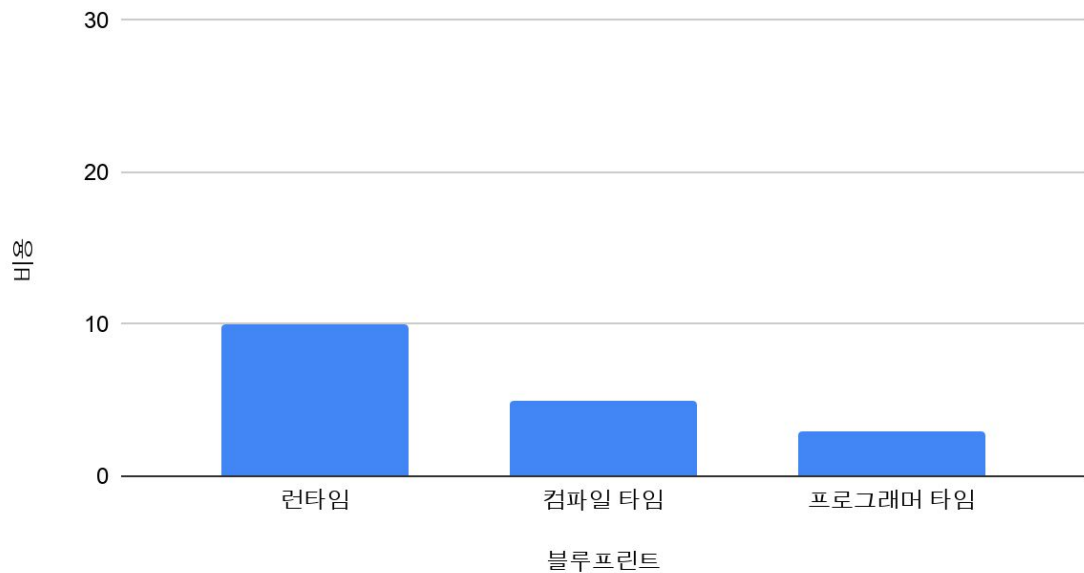
- 언리얼 cpp 를 할거임

블루프린트와 cpp 의 차이

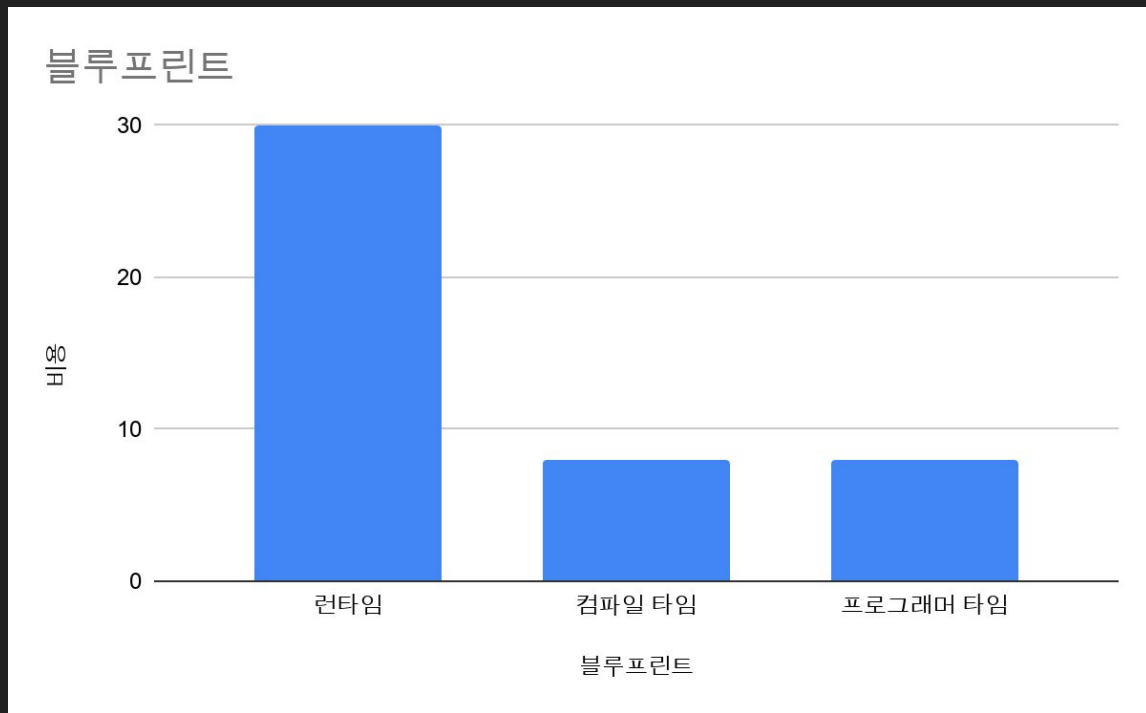
	CPP	블루프린트
속도	이론상 빠름	느림
코딩 난이도	높음	이론상 쉬움
한가지만 사용해서 가능한가?	가능은 함	많은부분이 가능하지만 안되는부분도 많음.

블루프린트 - 이상

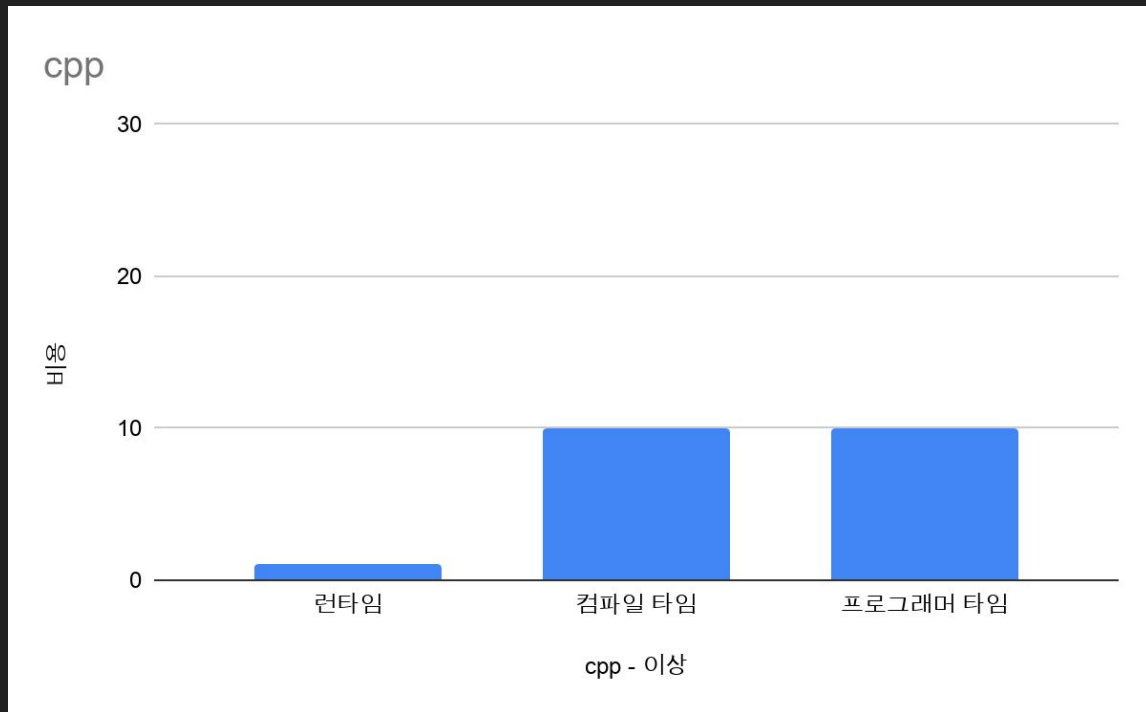
블루프린트



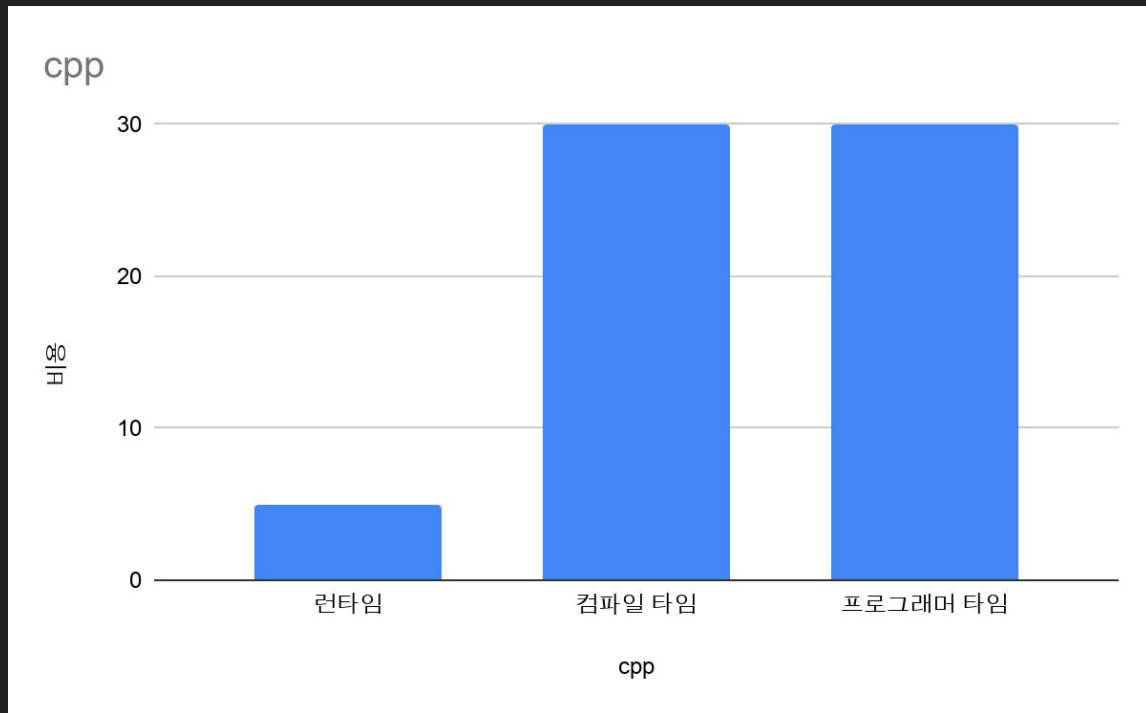
블루프린트 - 현실



CPP - 이상



CPP - 현실



그럼 왜 **CPP** 인가?

그럼 왜 C++ 인가?

- 어차피 회사라면 회사에서 쓰니까 써야됨.

블루프린트 몰라도 되는가?

- 근데 또 정작 가면 이거도 쓰긴 쓴다고 함.
- 그냥 할일이 2배!

실습

- 이론보다는 실습 위주로 갑니다.
- 이론은 필요할때 이야기 해드립니다.
- <https://docs.unrealengine.com/ko/Programming/Tutorials/PlayerInput/index.html> 를 기반으로 했습니다.

새 프로젝트

쉽죠.



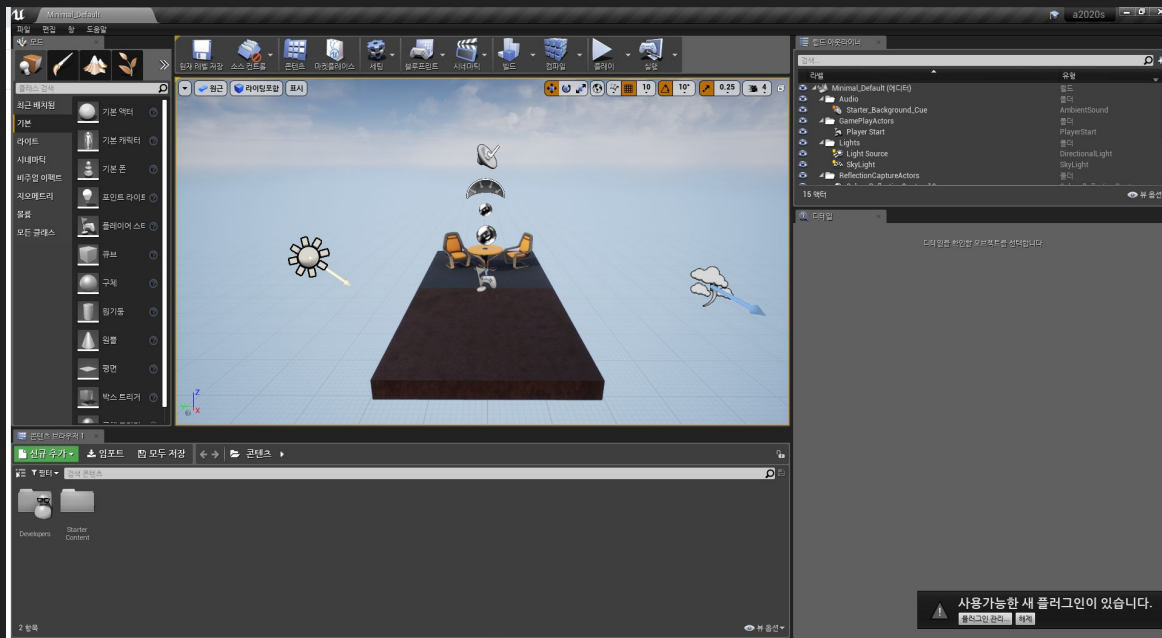
기본화면 - 대충설명.

화면 클릭하고 wasd
누르면 움직여짐.

화면 클릭하고 q 누르면
하강

화면 클릭하고 e 누르면
상승

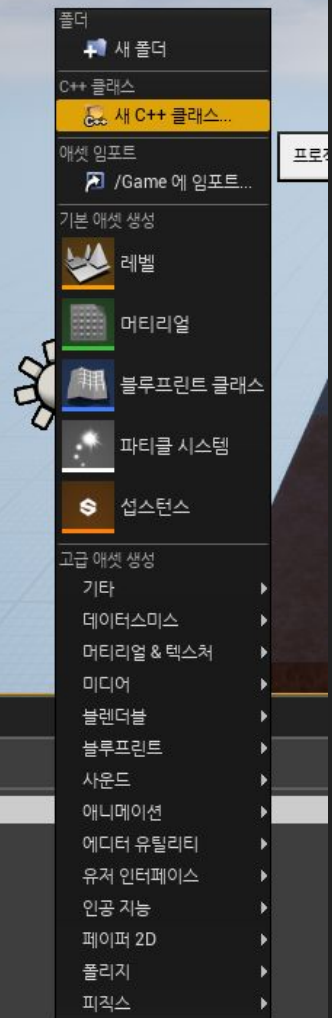
그냥 qwe 누르면 각각
이동 회전 스케일 모드



C++ 클래스 생성

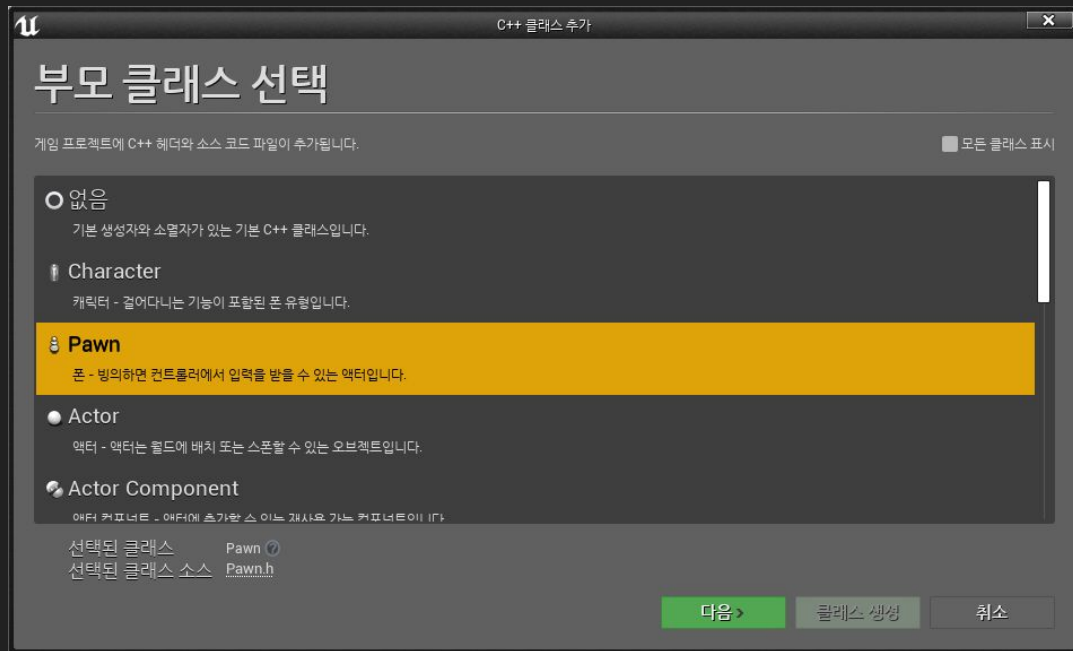
밑에 콘텐츠 브라우저에 오른쪽키 누르고

새 c++ 클래스.. 누르기



부모 클래스 선택

이번엔 pawn 으로 할거임



※ Pawn & Actor 설명

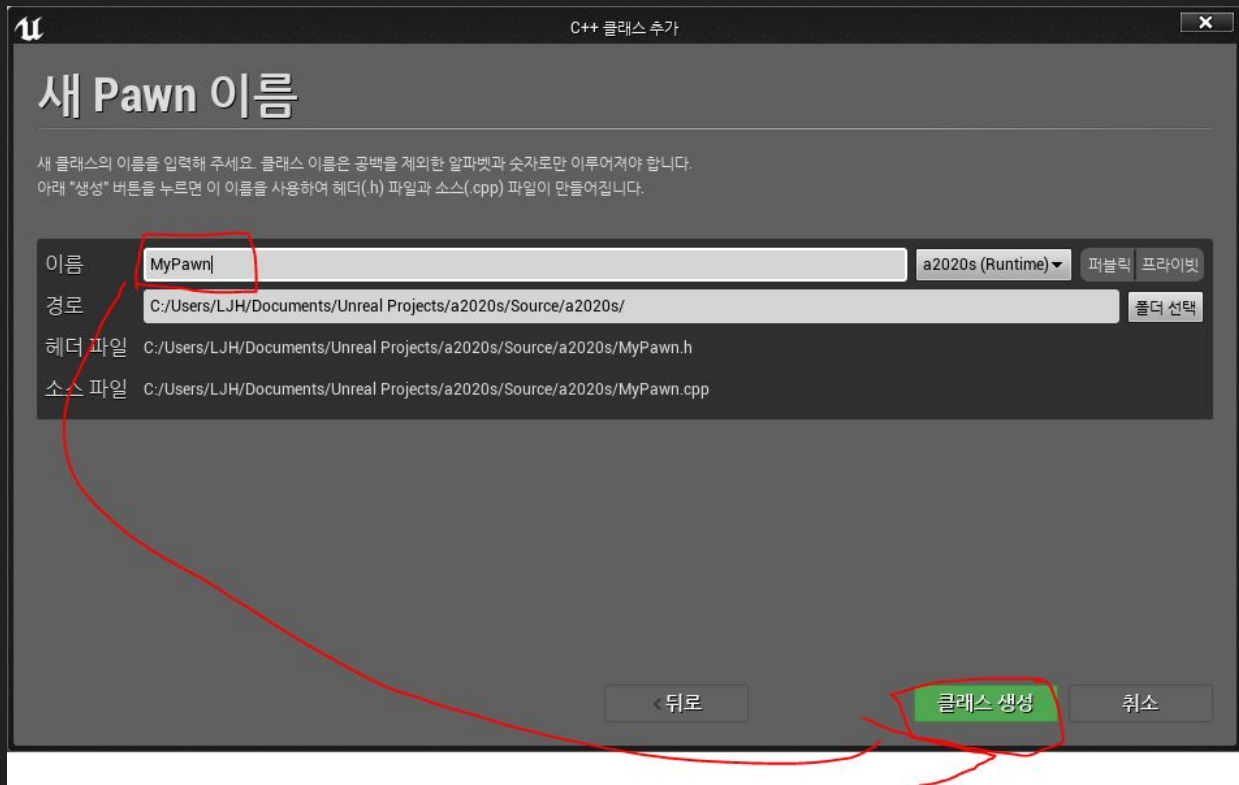
Pawn

- 가장 기본적인 플레이어가 움직일수 있는 캐릭터
- 인간형 캐릭터는 언리얼이 미리 만들어둔 `character` 를 상속하면 편함
-

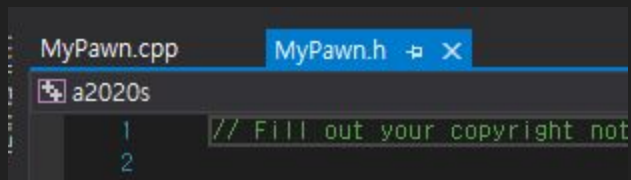
Actor

- 가장 기본적인 필드위에 있을수 있는 물체.

MyPawn 으로 생성



.h 랑 .cpp 가 생성됨



AMyPawn::AMypawn은 컴파일할때 실행됨

```
// Set default values
AMyPawn::AMyPawn( )
{
    // Set this pawn to call Tick() every frame. You can turn this off to improve performance if you don't need it.
    PrimaryActorTick.bCanEverTick = true;
}
```

BuginPlay

게임이 시작할때나 새로 생성되면 그때 호출됨

```
// Called when the game starts or when spawned
void AMyPawn::BeginPlay()
{
    Super::BeginPlay();
}
```

Tick

- 매 프레임마다 실행
- 성능에 매우 중대한 영향을 줌.

```
// Called every frame
void AMyPawn::Tick(float DeltaTime)
{
    Super::Tick(DeltaTime);
}
```

SetupPlayerInputComponent

인풋을 설정함.

```
// Called to bind functionality to input
void AMyPawn::SetupPlayerInputComponent(UInputComponent* PlayerInputComponent)
{
    Super::SetupPlayerInputComponent(PlayerInputComponent);
}
```

AMyPawn::AMyPawn() 에 추가

```
AMyPawn::AMyPawn()  
{  
    // Set this pawn to call Tick() every frame. You can turn this off to improve performance if you don't need it.  
    PrimaryActorTick.bCanEverTick = true;  
  
    // 이 폰을 가장 빠른 번호의 플레이어가 조종하도록 설정합니다  
    AutoPossessPlayer = EAutoReceiveInput::Player0;  
}
```

강제로 player0 (1인용 게임에서는 그냥 플레이어) 가 조정하게 하는 코드.

MyPawn.h 에 변수 추가.

```
public:
    // Called every frame
    virtual void Tick(float DeltaTime) override;

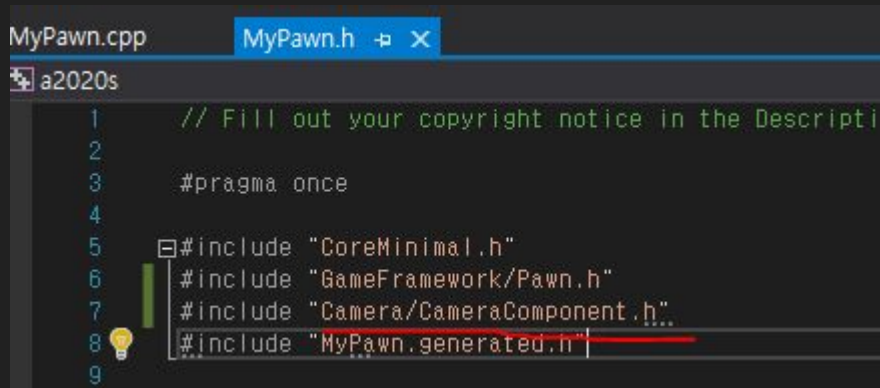
    // Called to bind functionality to input
    virtual void SetupPlayerInputComponent(class UInputComponent* PlayerInputComponent) override;

    UPROPERTY(EditAnywhere)
    USceneComponent* OurVisibleComponent;

};
```

MyPawn.h 에 추가.

무조건 Generated.h 위에
있어야함.



```
MyPawn.cpp  MyPawn.h  ×
a2020s
1 // Fill out your copyright notice in the Descripti
2
3 #pragma once
4
5 #include "CoreMinimal.h"
6 #include "GameFramework/Pawn.h"
7 #include "Camera/CameraComponent.h"
8 #include "MyPawn.generated.h"
9
```

다시 AMyPawn::AMyPawn() 에 추가

```
// Sets default values
AMyPawn::AMyPawn()
{
    // Set this pawn to call Tick() every frame. You can turn this off to improve performance if you don't need it.
    PrimaryActorTick.bCanEverTick = true;

    // 이 폰을 가장 빠른 번호의 플레이어가 조종하도록 설정합니다
    AutoPossessPlayer = EAutoReceiveInput::Player0;

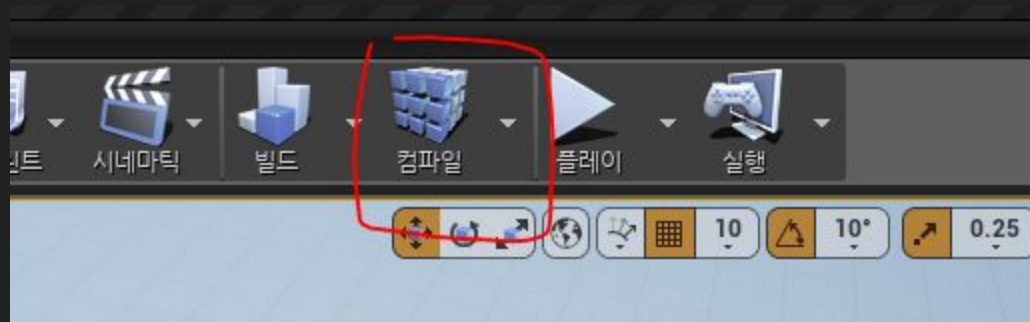
    // 무언가를 붙일 더미 루트 컴포넌트를 만듭니다
    RootComponent = CreateDefaultSubobject<USceneComponent>(TEXT("RootComponent"));
    // 카메라와 보이는 오브젝트를 만듭니다
    UCameraComponent* OurCamera = CreateDefaultSubobject<UCameraComponent>(TEXT("OurCamera"));
    OurVisibleComponent = CreateDefaultSubobject<UStaticMeshComponent>(TEXT("OurVisibleComponent"));
    // 루트 컴포넌트에 카메라와 보이는 오브젝트를 붙입니다. 카메라를 이격 및 회전시킵니다.
    OurCamera->SetupAttachment(RootComponent);
    OurCamera->SetRelativeLocation(FVector(-250.0f, 0.0f, 250.0f));
    OurCamera->SetRelativeRotation(FRotator(-45.0f, 0.0f, 0.0f));
    OurVisibleComponent->SetupAttachment(RootComponent);
}

// Called when the game starts or when spawned
```

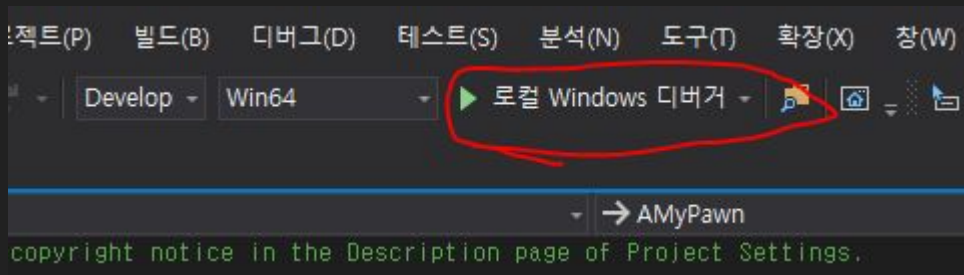
※ 빨간줄이 보이는건 기분탓이 아니다.

- 빨간줄은 있지만 빌드는 되는 이상한 일이 자주 생긴다.
- 익숙해지자.

컴파일을 누른다.



저러고 컴파일 되고도 안되는게 가끔 있는데



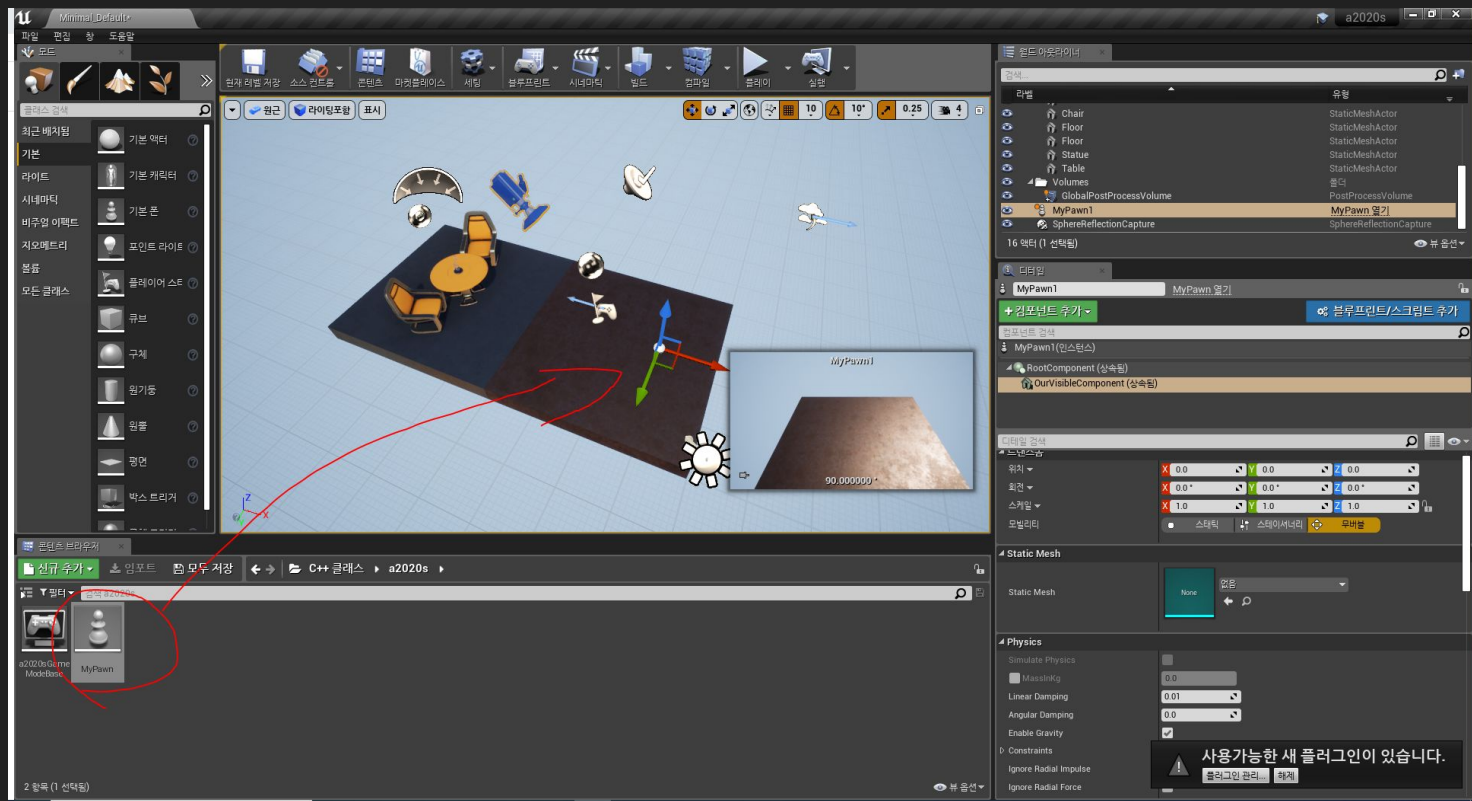
블루프린트랑 연동하는건 엔진 자체를 껐다 켜야된다.

이거를 누르면 다시 빌드 해서 엔진이 다시 켜진다.

그상태에선 된다.

그리고 엔진 싹다 껐다키면 된다.

드래이그 해서 드랍. 하면 필드위에 올려짐.

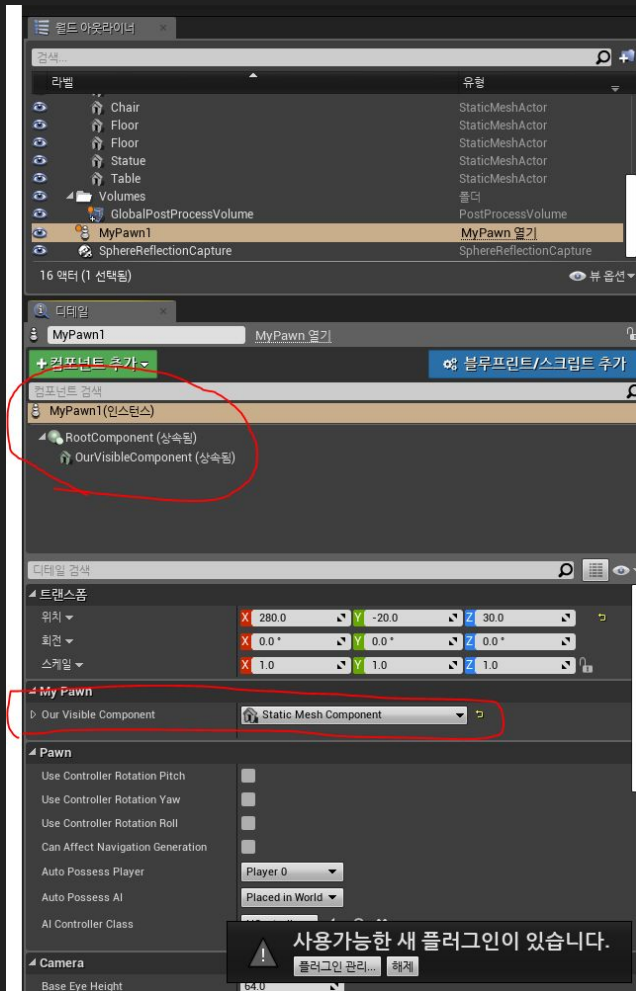


오른쪽에 MyPawn1 을 클릭

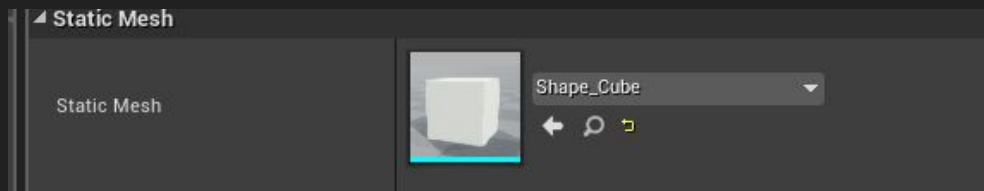
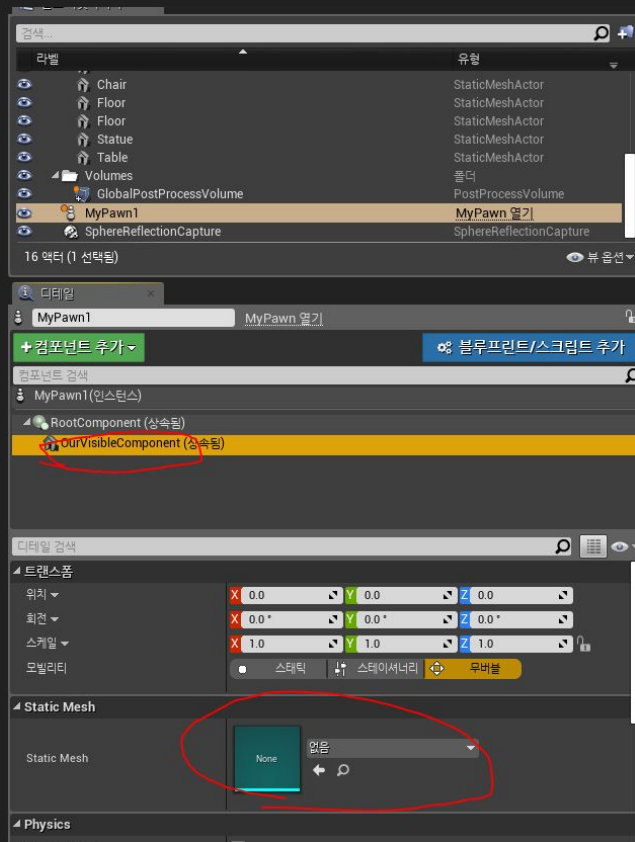
우리가 추가해준 컴포넌트와

변수가 추가되어있는걸 볼수있다.

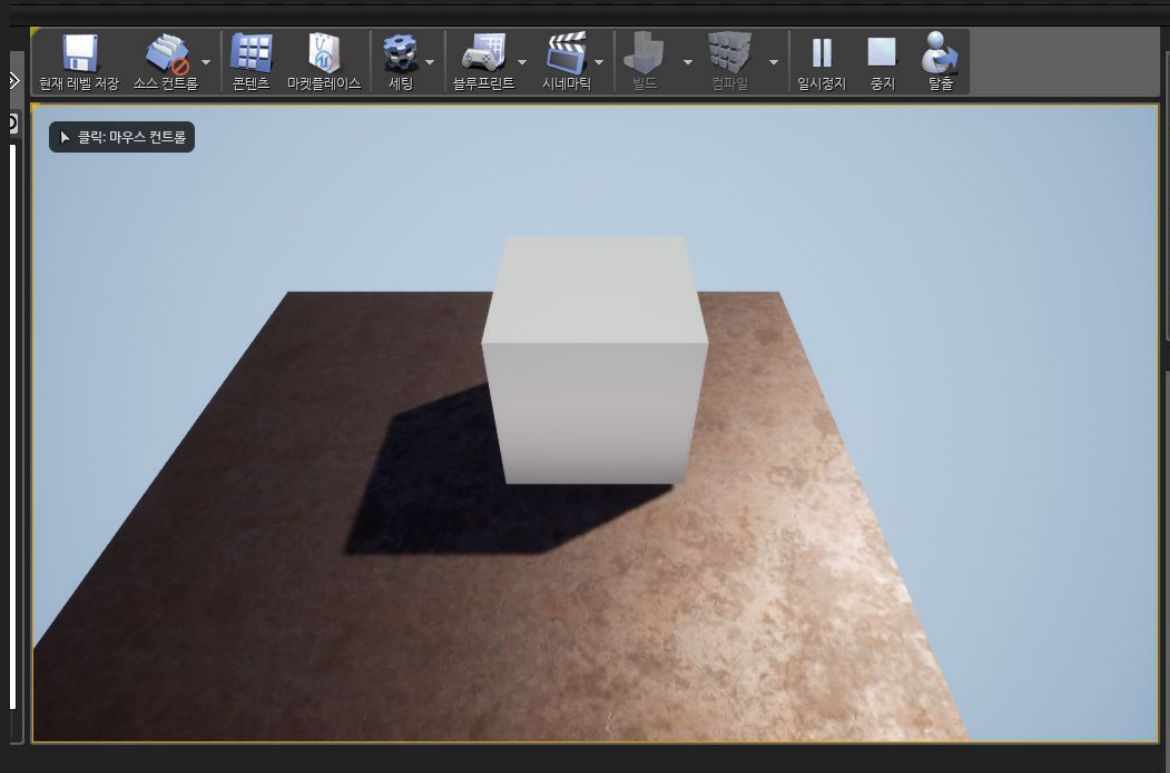
UPROPERTY(EditAnywhere)에
EditAnywhere 가 있기 때문에
변수를 수정해줄수 있다.



캐릭터 모양 만들어주기

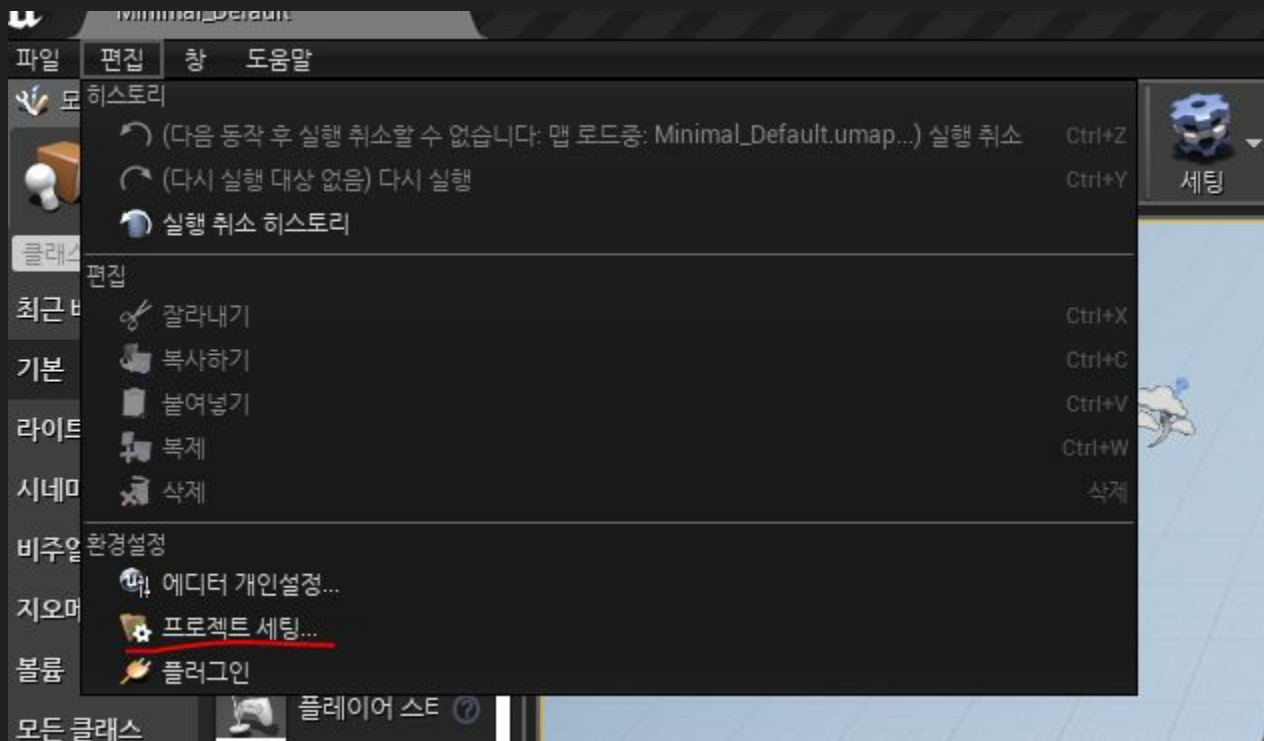


플레이를 눌러보면



입력받고 움직이게 하기

입력 설정 - 프로젝트 세팅



축 설정

일단 여기

모든 세팅

프로젝트

게임플레이 태그

멀 & 모드

무브

설정

알호환

지원 플랫폼

타겟 하드웨어

패키징

게임

엑셀 매니저

Asset Tools

엔진

가비지 콜렉션

게임플레이 디버거

계속할 LOD

내비게이션 메시

내비게이션 시스템

네트워크

렌더링

렌더링 오버라이드 (로컬)

스트리밍

슬라이트 세팅

애니메이션

오디오

유적 인터페이스

일반 세팅

입력

콘솔

클릭전

쿠러

디버깅 검색

엔진 - 입력

기본 입력 동작 및 축 설정이 포함된 입력 세팅입니다.

이 세팅은 DefaultInput.ini 에 저장되었으며, 현재 쓰기가 가능합니다.

Bindings

액션 및 축 매핑은 입력 동작과 그것을 실행하는 키 사이에 간접 충돌을 삽입하는 방식으로 입력 동작에 대한 키와 축 매핑을 편하게 할 수 있는 방법을 제공합니다. 액션 매핑은 키를 누르고 떼는 것, 축 매핑은 연속된 범위 입력에 대해

Action Mappings + -

Axis Mappings + -

Speech Mappings

0 배열 엘리먼트

+ -

Viewport Properties

Capture Mouse on Launch

☒

Default Viewport Mouse Capture Mode

Capture Permanently Including Initial Mouse Down

Default Viewport Mouse Lock Mode

Lock on Capture

Mobile

Always Show Touch Interface

☐

Show Console on Four Finger Tap

☒

Enable Gesture Recognizer

☒

Default Touch Interface



DefaultVirtualJoysticks

Virtual Keyboard (Mobile)

Use Autocorrect

☐

Console

Console Keys

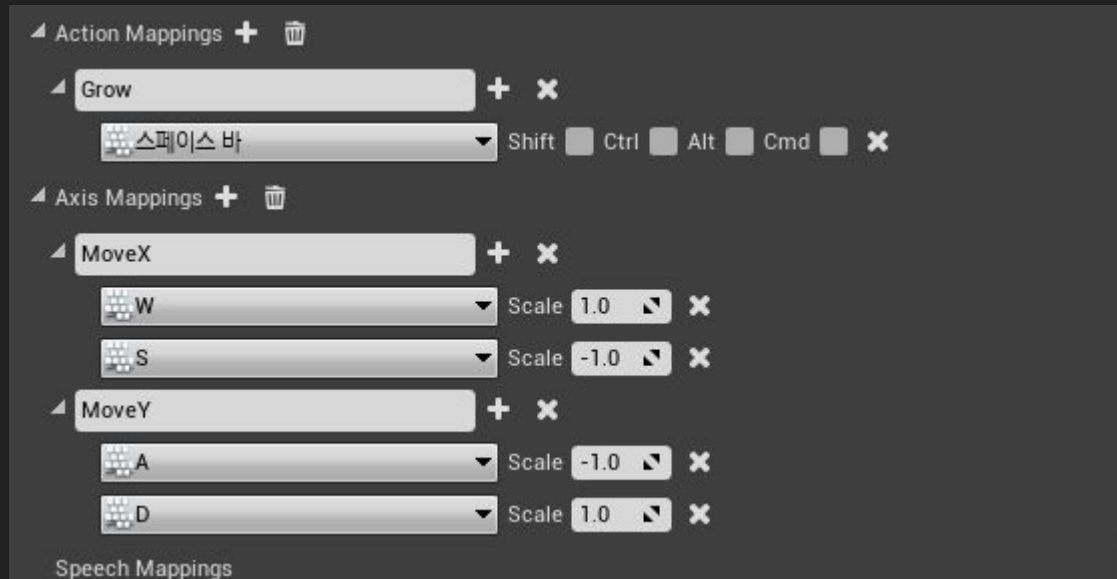
1 배열 엘리먼트

+ -

Mouse Properties

출입력 설정 추가.

Grow 는 좀있다 쓸거임.



MyPawn.h 에 추가

```
public:
    // Called every frame
    virtual void Tick(float DeltaTime) override;

    // Called to bind functionality to input
    virtual void SetupPlayerInputComponent(class UInputComponent* PlayerInputComponent) override;

    UPROPERTY(EditAnywhere)
    USceneComponent* OurVisibleComponent;

    //입력 함수
    void Move_XAxis(float AxisValue);
    void Move_YAxis(float AxisValue);

    //입력 함수
    FVector CurrentVelocity;
```

MyPawn.cpp 에 구현

Clamp 는 지정된 값을 벗어나면 그 안으로 꾸겨넣는다.

```
void AMyPawn::Move_XAxis(float AxisValue)
{
    // 초당 100 유닛을 앞 또는 뒤로 움직입니다
    CurrentVelocity.X = FMath::Clamp(AxisValue, -1.0f, 1.0f) * 100.0f;
}

void AMyPawn::Move_YAxis(float AxisValue)
{
    // 초당 100 유닛을 오른쪽 또는 왼쪽으로 움직입니다
    CurrentVelocity.Y = FMath::Clamp(AxisValue, -1.0f, 1.0f) * 100.0f;
}
```


Tick 에 추가

프레임마다
실제로 움직이게
한다.

```
// Called every frame
void AMyPawn::Tick(float DeltaTime)
{
    Super::Tick(DeltaTime);

    if (!CurrentVelocity.IsZero())
    {
        FVector NewLocation = GetActorLocation() + (CurrentVelocity * DeltaTime);
        SetActorLocation(NewLocation);
    }
}
```

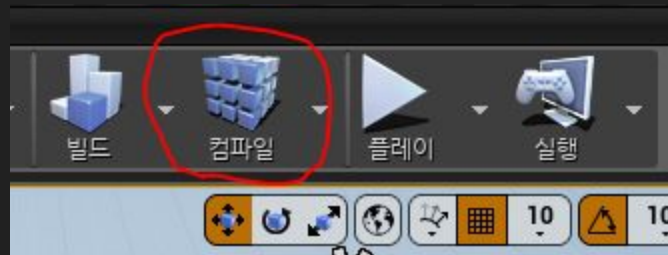
```
// ... and the corresponding ...
```

SetupPlayerInputComponent 에 추가

```
// Called to bind functionality to input
void AMyPawn::SetupPlayerInputComponent(UInputComponent* PlayerInputComponent)
{
    Super::SetupPlayerInputComponent(PlayerInputComponent);

    // "MoveX" 와 "MoveY" 두 이동 축의 값에 매 프레임 반응합니다
    PlayerInputComponent->BindAxis("MoveX", this, &AMyPawn::Move_XAxis);
    PlayerInputComponent->BindAxis("MoveY", this, &AMyPawn::Move_YAxis);
}
```

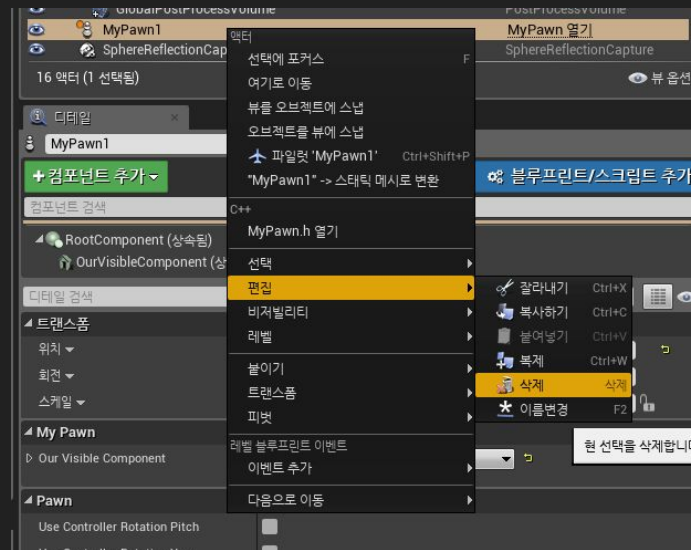
또 컴파일



원래 필드에 있는거 삭제

삭제하고

다시 MyPawn dragged 해서 올리기



해보자

움직인다.

SpaceBar 를 누르면 커지게 만들기

MyPawn.h 에 추가

```
//입력 함수  
void Move_XAxis(float AxisValue);  
void Move_YAxis(float AxisValue);  
void StartGrowing();  
void StopGrowing();  
  
//입력 함수  
FVector CurrentVelocity;  
bool bGrowing;
```

구현

```
void AMyPawn::StartGrowing()  
{  
    bGrowing = true;  
}
```

```
void AMyPawn::StopGrowing()  
{  
    bGrowing = false;  
}
```


SetupPlayerInputComponent 추가

```
// Called to bind functionality to input
void AMyPawn::SetupPlayerInputComponent(UInputComponent* PlayerInputComponent)
{
    Super::SetupPlayerInputComponent(PlayerInputComponent);

    // "MoveX" 와 "MoveY" 두 이동 축의 값에 매 프레임 반응합니다
    PlayerInputComponent->BindAxis("MoveX", this, &AMyPawn::Move_XAxis);
    PlayerInputComponent->BindAxis("MoveY", this, &AMyPawn::Move_YAxis);

    // "Grow" 키를 누르거나 땔 때 반응합니다
    PlayerInputComponent->BindAction("Grow", IE_Pressed, this, &AMyPawn::StartGrowing);
    PlayerInputComponent->BindAction("Grow", IE_Released, this, &AMyPawn::StopGrowing);
}
```

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

AMyPawn::Tick 에 추가

```
// Called every frame
void AMyPawn::Tick(float DeltaTime)
{
    Super::Tick(DeltaTime);

    if (!CurrentVelocity.IsZero())
    {
        FVector NewLocation = GetActorLocation() + (CurrentVelocity * DeltaTime);
        SetActorLocation(NewLocation);
    }

    float CurrentScale = OurVisibleComponent->GetComponentScale().X;
    if (bGrowing)
    {
        // 1 초에 걸쳐 두 배 크기로 키웁니다
        CurrentScale += DeltaTime;
    }
    else
    {
        // 키운 속도대로 절반으로 줄입니다
        CurrentScale -= (DeltaTime * 0.5f);
    }

    // 시작 크기 아래로 줄이거나 두 배 이상으로 키우지 않도록 합니다.
    CurrentScale = FMath::Clamp(CurrentScale, 1.0f, 2.0f);
    OurVisibleComponent->SetWorldScale3D(FVector(CurrentScale));
}
```

또 컴파일

커진다

