


국민대학교 2019 캡스톤 프로젝트

LOST DARK

Surreal (16조) – 김성훈, 김종균, 송영륜, 장윤지, 갈경달

중간 보고



국민대학교 2019 캡스톤 프로젝트

LOST DARK

Surreal (16조) – 김성훈, 김종균, 송영륜, 장윤지, 갈경달

중간 보고

연표

<https://drive.google.com/file/d/1vADcS-iaK3iN3sU86u2rAwlOPbKkKI6Y/view?usp=sharing>

INDEX

1. Lost Dark란?
2. 게임 기획 의도
3. Art Concept
4. 수행 내용 및 기술 설명
5. 향후 계획
6. Q/A

Lost Dark란?

SYNOPSIS

부상당한 주인공은 A마을에서 기억상실의 상태로 깨어난다.
당시 옆 마을 B는 그리폰(보스 몬스터)로 인해 폭파당한 상태이며
그리폰이 사는 곳에는 강한 크리스탈도 있다는 것을 알게 된다.

마을 사람들은 자원에 대한 욕심으로 토벌대를 조직하여 그곳으로 모험을 떠나는데,
이에 주인공도 참여하며 게임이 진행된다.

Lost Dark란?

장르

어드벤처 RPG

플랫폼

PC

그래픽

High poly 3D
3인칭 Back view

게임 기획 의도

목표

- 직관적이고 자유로운 조작 방식과 타격감 넘치는 전투 시스템, 그리고 흥미로운 시나리오를 통해 사용자에게 **재미를 준다.**
- 게임을 진행하며 플레이어가 생각하는 가치에 근거하여 선택을 해 나가야 하는 상황들을 통해, **개인이 살아가며 추구해야 할 가치를 되돌아 보는 시간**을 갖게 된다. 이는 내면 정비에 도움을 주어 게임의 재미 요소와 함께 **플레이어의 스트레스 경감에** 도움을 준다.

프로젝트 진행

The screenshot displays a Trello board for 'Unreal 캡스톤 디자인' (Unreal Capstone Design). The board is organized into several columns representing different stages of the project:

- 공지사항 & 전달사항** (Notice & Delivery): Contains cards for '4월 3주차 일정' (4th week schedule), '역할 분담 및 실제 진행과정' (Role assignment and actual progress), '캡스톤 평가 기준 목록' (Capstone evaluation criteria), and '폴더관리 규칙' (Folder management rules). A card titled '공통' (Common) is also visible.
- 해야하는 작업들(To do list)** (Things to do): Includes a card for '유튜브 튜토리얼 강좌 5개(추가중)' (5 YouTube tutorial lectures (adding more)).
- 개발 하는중(Doing)** (Working on development): Lists tasks such as '[중간발표] 계획서 수정' (Intermediate presentation plan revision), '[중간발표] PPT 만들기' (Intermediate presentation PPT making), '[중간발표] Github 수정작업' (Intermediate presentation Github update work), '[중간발표] 시연 영상' (Intermediate presentation demo video), '[중간발표] 중간발표 문서' (Intermediate presentation document), and 'Github 작업할 부분 (끝까지)' (Github work parts (until the end)).
- 개발 완료(Be done)** (Development completed): Lists completed tasks like '수정)' (Revision), 'Text window UI', '캐릭터 회피' (Character evasion), '몬스터-플레이어 추적 (몬스터 AI)' (Monster-player tracking (Monster AI)), '[C++] 카메라 회전 두가지를 혼합한 경우(로스트아크+배그)' (Mixed camera rotation cases (Lost Ark + BGG)), '[C++] 카메라 회전(로스트아크, 디아블로 방식)' (Camera rotation (Lost Ark, Diablo style)), and '[C++] 카메라 회전(GTA=배틀그라운드)' (Camera rotation (GTA = Battle Royale)).
- 개발 아이디어 & 의견 & 각종 Tip** (Development ideas & opinions & various tips): Includes cards for 'Unreal 코딩 표준' (Unreal coding standards), '★★유니티 개발자를 위한 언리얼 이해 Tip' (Unreal understanding tip for Unity developers), '심리학적 근거 - 참고자료' (Psychological basis - reference material), '언리얼 할때 Visual Studio 설정 Tip' (Unreal Visual Studio settings tip), '언리얼 공식 카페 (정보많음)' (Unreal official cafe (lots of info)), and '순서도 그리기 - 참고' (Flowchart drawing - reference).
- 회의록(Meeting Minutes)** (Meeting minutes): Contains a series of cards detailing meeting minutes, including dates and topics like '7회차 [기술] 4월 11일 (목) 회의' (7th session [Tech] April 11th (Thu) meeting), '6회차 [기획] 4월 9일 (화) 회의' (6th session [Planning] April 9th (Tue) meeting), '5회차 [개발] 4월 6일 (토) 회의' (5th session [Development] April 6th (Sat) meeting), and '4회차 [기획] 4월 2일 (화) 회의' (4th session [Planning] April 2nd (Tue) meeting).

Art Concept



Art Concept

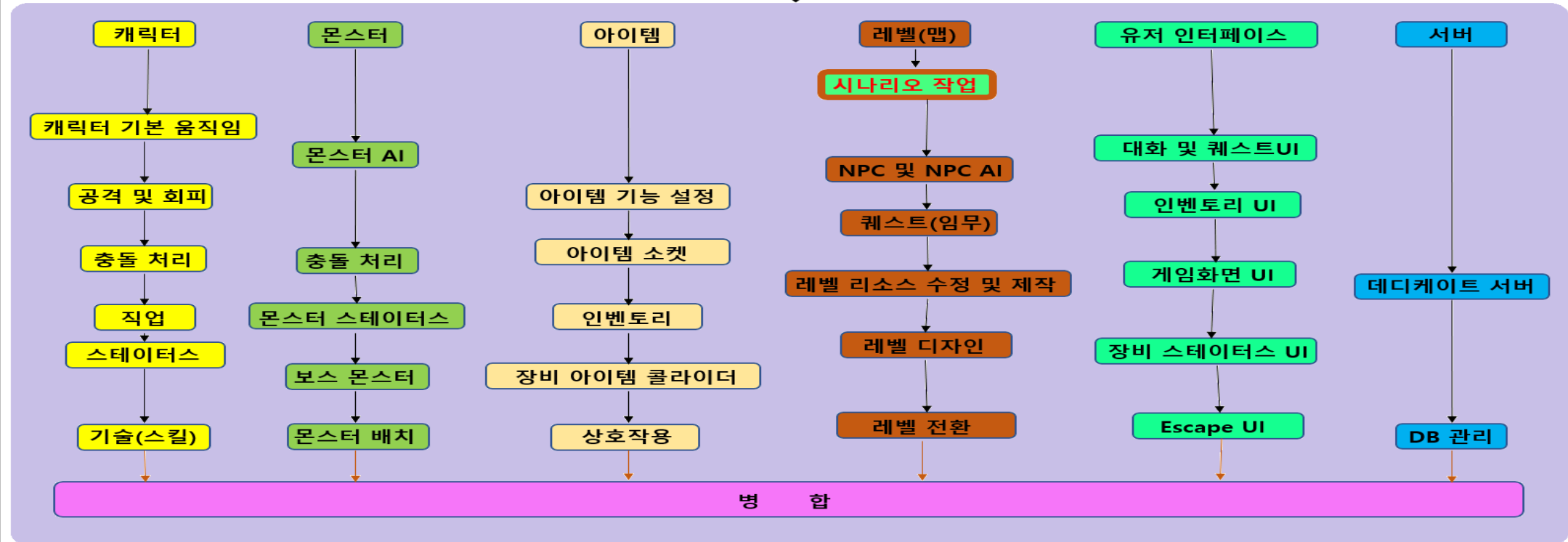


Art Concept



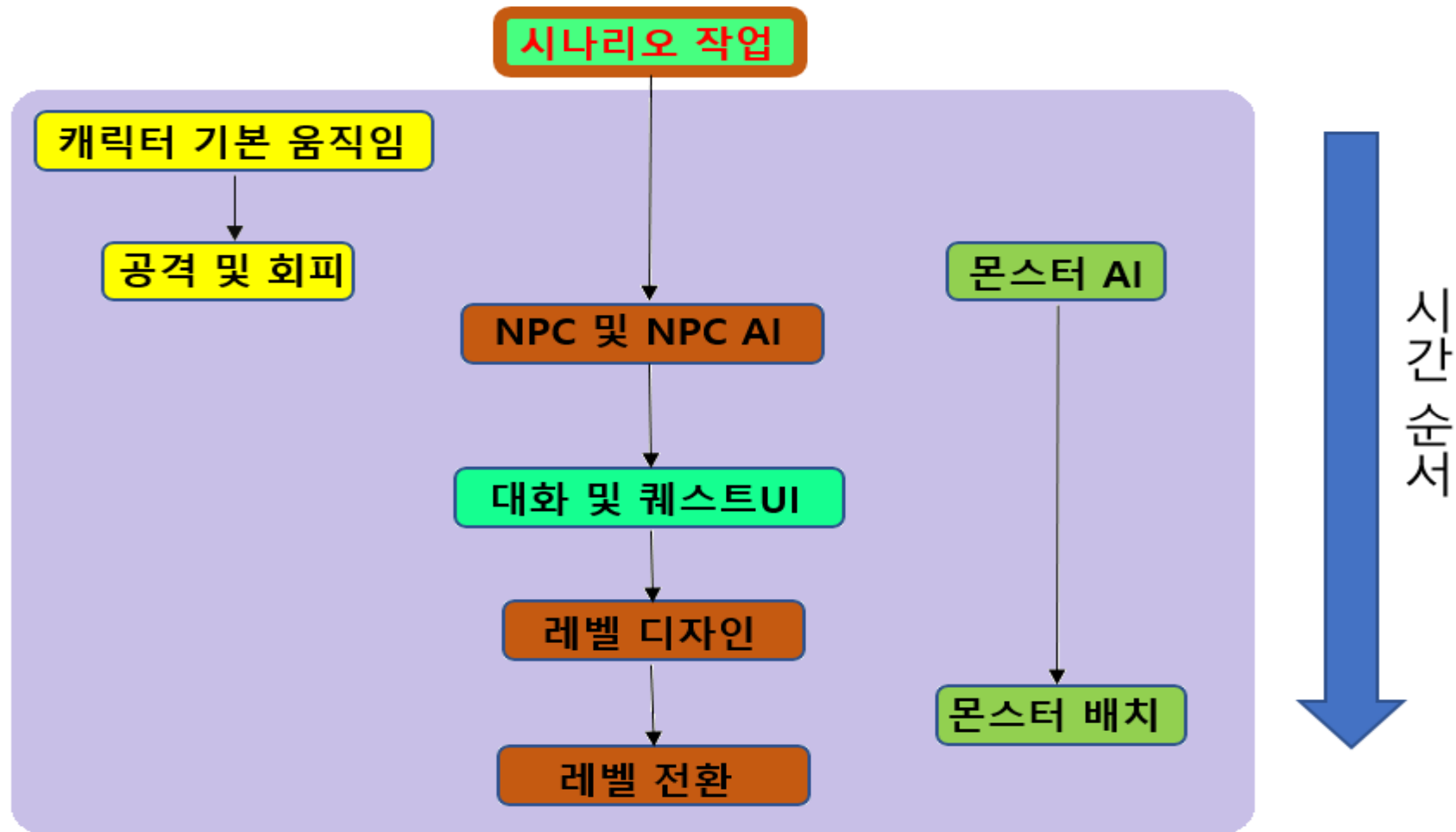
절차도

개발 시작



개발 완료

수행 내용



캐릭터 이동

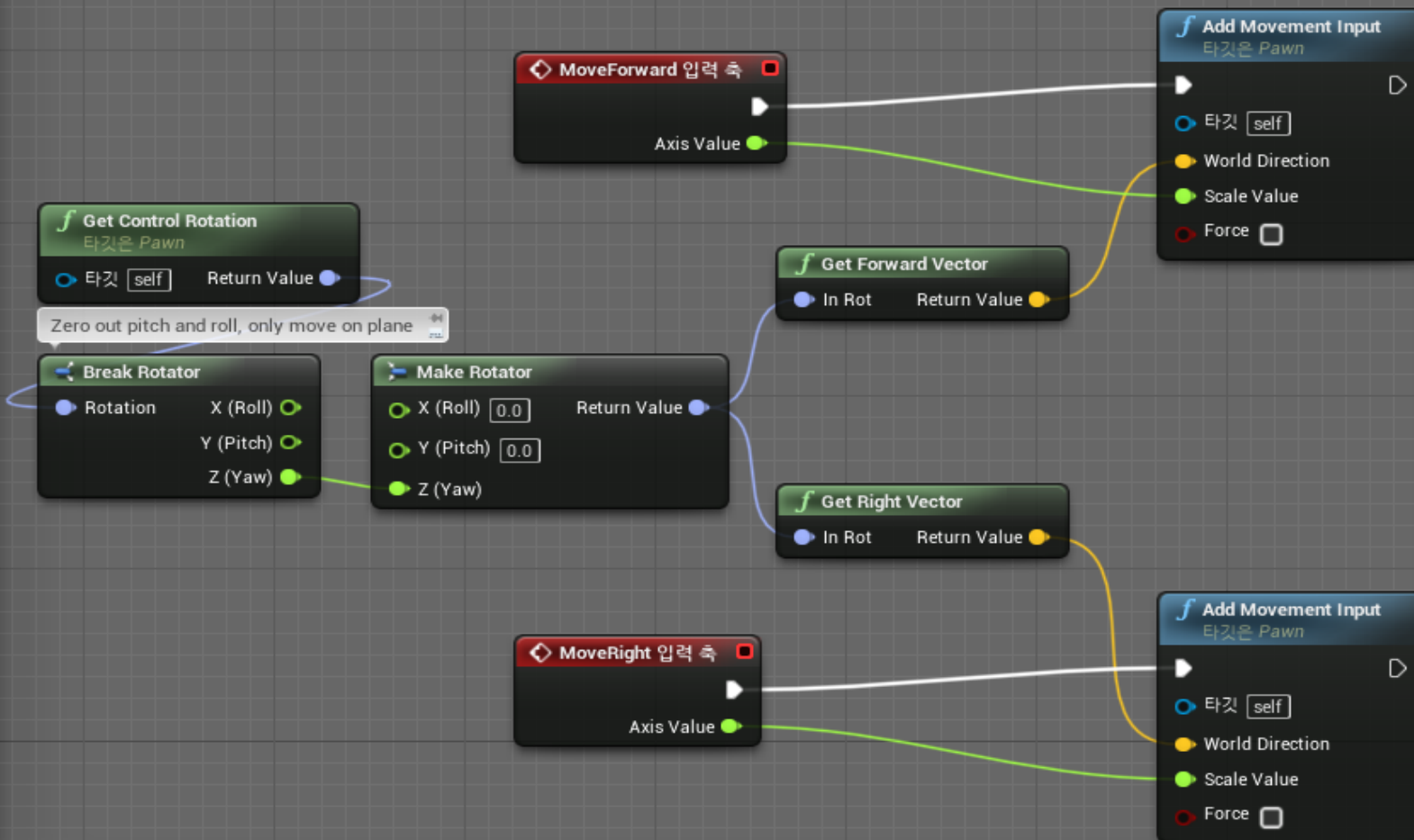
캐릭터 기본 움직임

공격 및 회피

레벨 전환

대화 및 퀘스트UI

Movement input



캐릭터 이동

```
ABPawn.h  ABPlayerController.h  ABCharacter.h  ABAnimInstance.h  ABGameMode.h  ABPawn.cpp  ABPlayerController.cpp  ABCharacter.cpp  ABAnimInstance.cpp  ABGameMode.cpp
ArenaBattle.h  AABCharacter
1 // Fill out your copyright notice in the Description page of Project Settings.
2
3 #pragma once
4
5 #include "ArenaBattle.h"
6 #include "GameFramework/Character.h"
7 #include "ABCharacter.generated.h"
8
9 UCLASS()
10 class ARENABATTLE_API AABCharacter : public ACharacter
11 {
12     GENERATED_BODY()
13
14 public:
15     // Sets default values for this character's properties
16     AABCharacter();
17
18 protected:
19     // Called when the game starts or when spawned
20     virtual void BeginPlay() override;
21
22     // 새로 추가
23     enum class EControlMode
24     {
25         GTA,
26         DIABLO
27     };
28     // 컨트롤 모드 설정 함수
29     void SetControlMode(EControlMode NewControlMode);
30     // 열거형 변수 선언 및 초기화 (기본 GTA)
31     EControlMode CurrentControlMode = EControlMode::GTA;
32     // 방향 앞방향 시선값. UPROPERTY를 사용하지 않는 FVector는 항상 초기값을 미리 지정해야 안전하다.
33     // 만약 축 입력 이벤트가 발생하면, 해당 변수를 업데이트하고 Tick로직에서 이것을 이용해 처리한다.
34     FVector DirectionToMove = FVector::ZeroVector;
35
36     // 시점 변경할때 사용할 변수들
37     float ArmLengthTo = 0.0f;
38     FRotator ArmRotationTo = FRotator::ZeroRotator;
39     float ArmLengthSpeed = 0.0f;
40     float ArmRotationSpeed = 0.0f;
41
42 public:
43     // Called every frame
44     virtual void Tick(float DeltaTime) override;
45
46     // PostInitializeComponents 컴포넌트 초기화 이후, 액티비티 컴포넌트 초기화 완료 후 호출됩니다.
47
48 ArenaBattle.cpp
49 AABCharacter
50 1 // Fill out your copyright notice in the Description page of Project Settings.
51 2
52 #include "ABCharacter.h"
53 // 공격 애니메이션을 재생하라는 명령을 넣기 위해서.
54 #include "ABAnimInstance.h"
55 // 디버깅드로잉
56 #include "DrawDebugHelpers.h"
57
58 // Sets default values
59 AABCharacter::AABCharacter()
60 {
61     // Set this character to call Tick() every frame. You can turn this off to improve performance if
62     PrimaryActorTick.bCanEverTick = true;
63
64     // 컴포넌트 추가
65     SpringArm = CreateDefaultSubobject<USpringArmComponent>(TEXT("SPRINGARM"));
66     Camera = CreateDefaultSubobject<UCameraComponent>(TEXT("CAMERA"));
67
68     // 컴포넌트를 상속시킴.
69     SpringArm->SetupAttachment(GetCapsuleComponent());
70     Camera->SetupAttachment(SpringArm);
71
72     GetMesh()->SetRelativeLocationAndRotation(FVector(0.0f, 0.0f, -88.0f), FRotator(0.0f, -90.0f, 0.0f));
73     SpringArm->TargetArmLength = 400.0f;
74     SpringArm->SetRelativeRotation(FRotator(-15.0f, 0.0f, 0.0f));
75
76     static ConstructorHelpers::FObjectFinder<USkeletalMesh> SK_CARDBOARD(TEXT("/Game/InfinityBladeWarr...
77
78     if (SK_CARDBOARD.Succeeded())
79     {
80         GetMesh()->SetSkeletalMesh(SK_CARDBOARD.Object);
81     }
82
83     GetMesh()->SetAnimationMode(EAnimationMode::AnimationBlueprint);
84
85     static ConstructorHelpers::FClassFinder<UAnimInstance> WARRIOR_ANIM(TEXT("/Game/Book/Animations/War...
86
87     if (WARRIOR_ANIM.Succeeded())
88     {
89         GetMesh()->SetAnimInstanceClass(WARRIOR_ANIM.Class);
90     }
91
92     // 최초 기본 모드는 DIABLO형식이다.
93     SetControlMode(EControlMode::DIABLO);
94
95     ArmLengthSpeed = 3.0f;
```

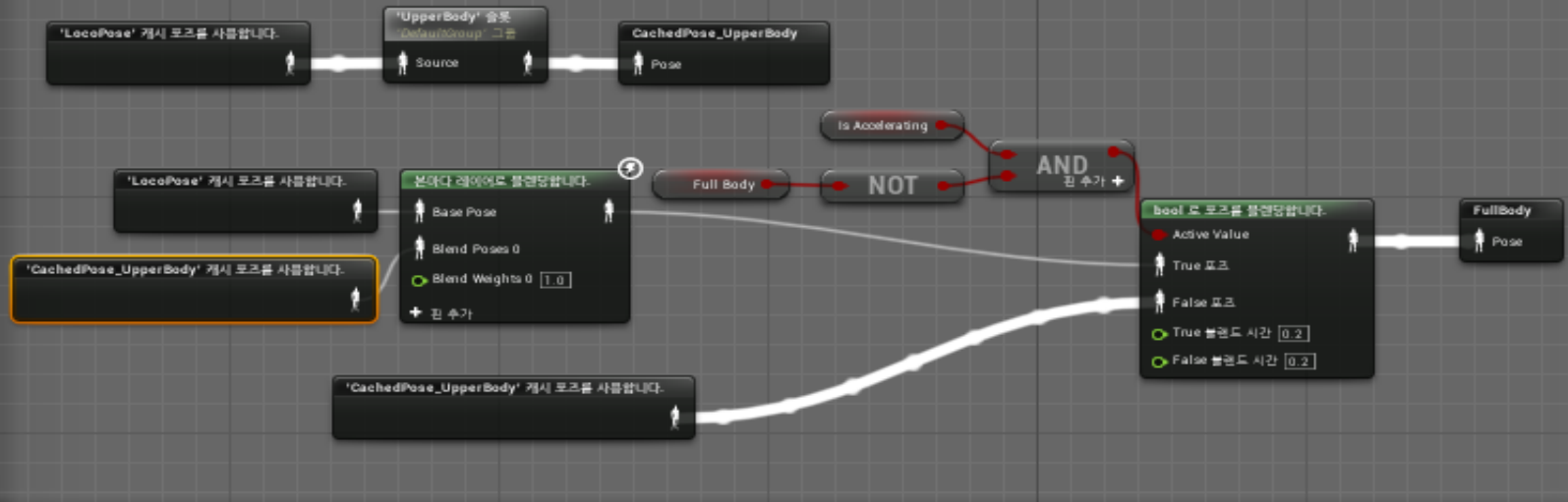
캐릭터 이동



애니메이션

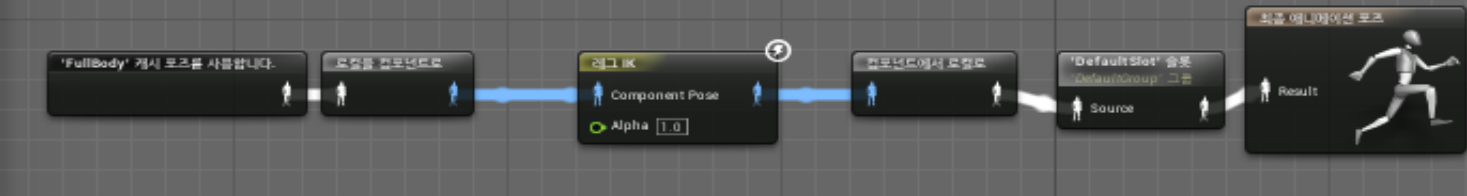
Upper Body Layer

Upper Body Layer

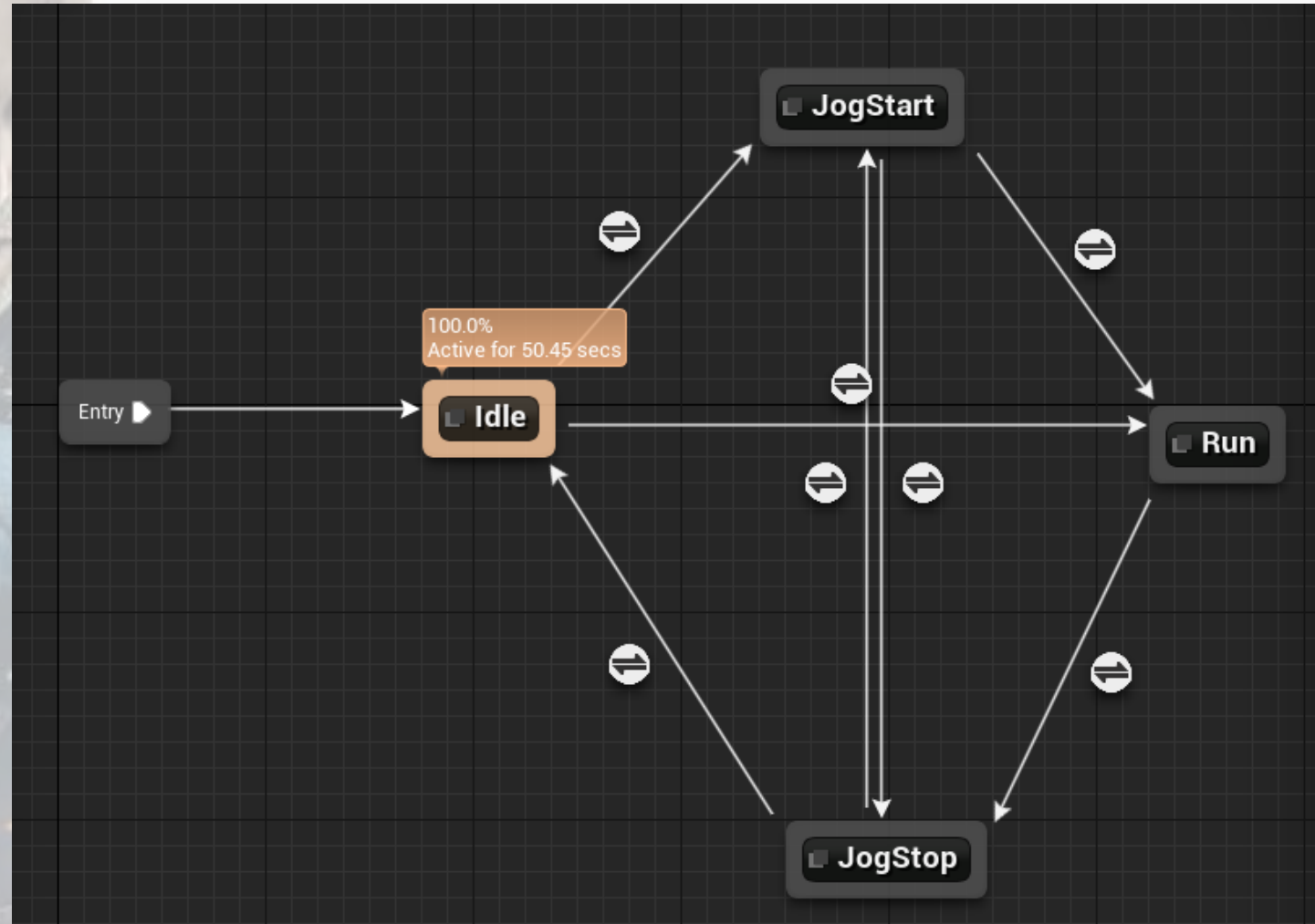


Foot IK controls

Foot IK controls



애니메이션

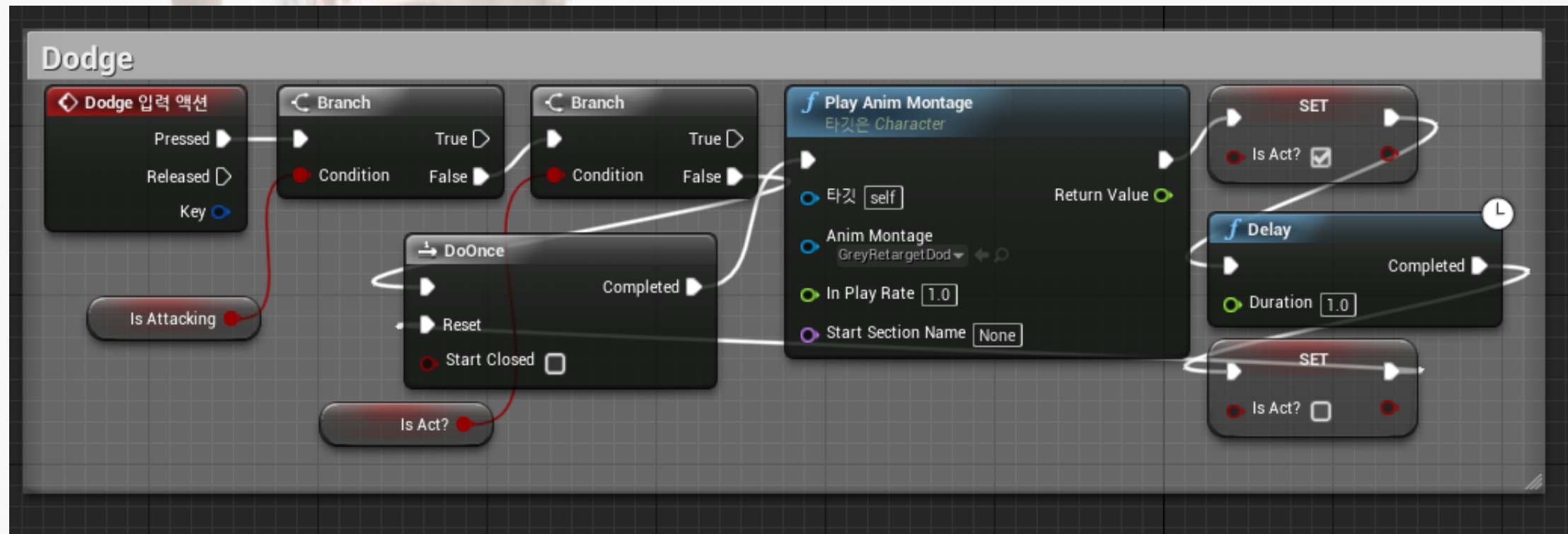


애니메이션

```
ABPawn.h  ABPlayerController.h  ABCharacter.h  ABAnimInstance.h  ABGameMode.h
ArenaBattle.h
ArenaBattle (전역 범위) DECLARE_MULTICAST_DELEGATE(FOnNextAttackCheckDelegate);
1 // Fill out your copyright notice in the Description page of Project Settings.
2
3 #pragma once
4
5 #include "ArenaBattle.h"
6 #include "Animation/AnimInstance.h"
7 #include "ABAnimInstance.generated.h"
8
9 // 멀티캐스트 델리게이트 선언
10 DECLARE_MULTICAST_DELEGATE(FOnNextAttackCheckDelegate);
11 DECLARE_MULTICAST_DELEGATE(FOnAttackHitCheckDelegate);
12
13 /**
14  *
15  */
16 UCLASS()
17 class ARENABATTLE_API UABAnimInstance : public UAnimInstance
18 {
19     GENERATED_BODY()
20
21 public:
22     // 생성자 만들어주고
23     UABAnimInstance();
24     // tick 마다 호출하는 가상함수 선언.
25     virtual void NativeUpdateAnimation(float DeltaSeconds) override;
26     // 몽타주 플레이하는 함수
27     void PlayAttackMontage();
28
29     // 다음 몽타주 섹션으로 점프하는 함수
30     void JumpToAttackMontageSection(int32 NewSection);
31
32 public:
33     // 델리게이트
34     FOnNextAttackCheckDelegate OnNextAttackCheck;
35     FOnAttackHitCheckDelegate OnAttackHitCheck;
36
37     // 죽는 애니메이션 설정
38     void SetDeadAnim() { IsDead = true; }
39
40 private:
41     // 노티파이를 위한 함수.
42     UFUNCTION()
43     void AnimNotify_AttackHitCheck();
44
45     // 다음 공격 체크 노티파이
46     UFUNCTION()
```

```
ABPawn.cpp  ABPlayerController.cpp  ABCharacter.cpp  ABAnimInstance.cpp  ABGameMode.cpp
ArenaBattle.cpp
ArenaBattle UABAnimInstance NativeUpdateAnimation(float DeltaSeconds)
1 // Fill out your copyright notice in the Description page of Project Settings.
2
3 #include "ABAnimInstance.h"
4
5 // 생성자
6 UABAnimInstance::UABAnimInstance()
7 {
8     // 초기 속도는 0이다
9     CurrentPawnSpeed = 0.0f;
10    // 초기 공중 상태는 false다.
11    IsInAir = false;
12    // 초기 죽었는지 변수는 false;
13    IsDead = false;
14
15    // 애니메이션 몽타주 등록
16    static ConstructorHelpers::FObjectFinder<UAnimMontage> ATTACK_MONTAGE(TEXT("/Game/Book/Animations/"));
17    // 경로에 애니메이션 몽타주가 유효하다면,
18    if (ATTACK_MONTAGE.Succeeded())
19    {
20        // 애니메이션 몽타주 변수에 해당 몽타주 애니메이션 정보를 등록한다.
21        AttackMontage = ATTACK_MONTAGE.Object;
22    }
23 }
24
25 // tick마다 호출되는 함수. 즉, Tick과 거의 동일하다고 봐도 무방할듯.
26 void UABAnimInstance::NativeUpdateAnimation(float DeltaSeconds)
27 {
28     Super::NativeUpdateAnimation(DeltaSeconds);
29
30     // Pawn 객체 생성, 폰에 접근해 속력값을 가져올때 사용함.
31     auto Pawn = TryGetPawnOwner();
32
33     // 폰에 접근하지 못했던만 반환
34     if (!IsValid(Pawn)) return;
35
36     // 죽지 않았다면
37     if (!IsDead)
38     {
39         // 애니메이션스턴스의 프로퍼티(멤버변수)값에 폰의 현재속력을 업데이트 시킨다.
40         CurrentPawnSpeed = Pawn->GetVelocity().Size();
41
42         // 임시 캐릭터 변수를 만들어 우리가 만든 캐릭터 클래스 정보를 넘겨주고,
43         auto Character = Cast<ACharacter>(Pawn);
44         // 캐릭터가 유효하다면,
45         if (Character)
46         {
```

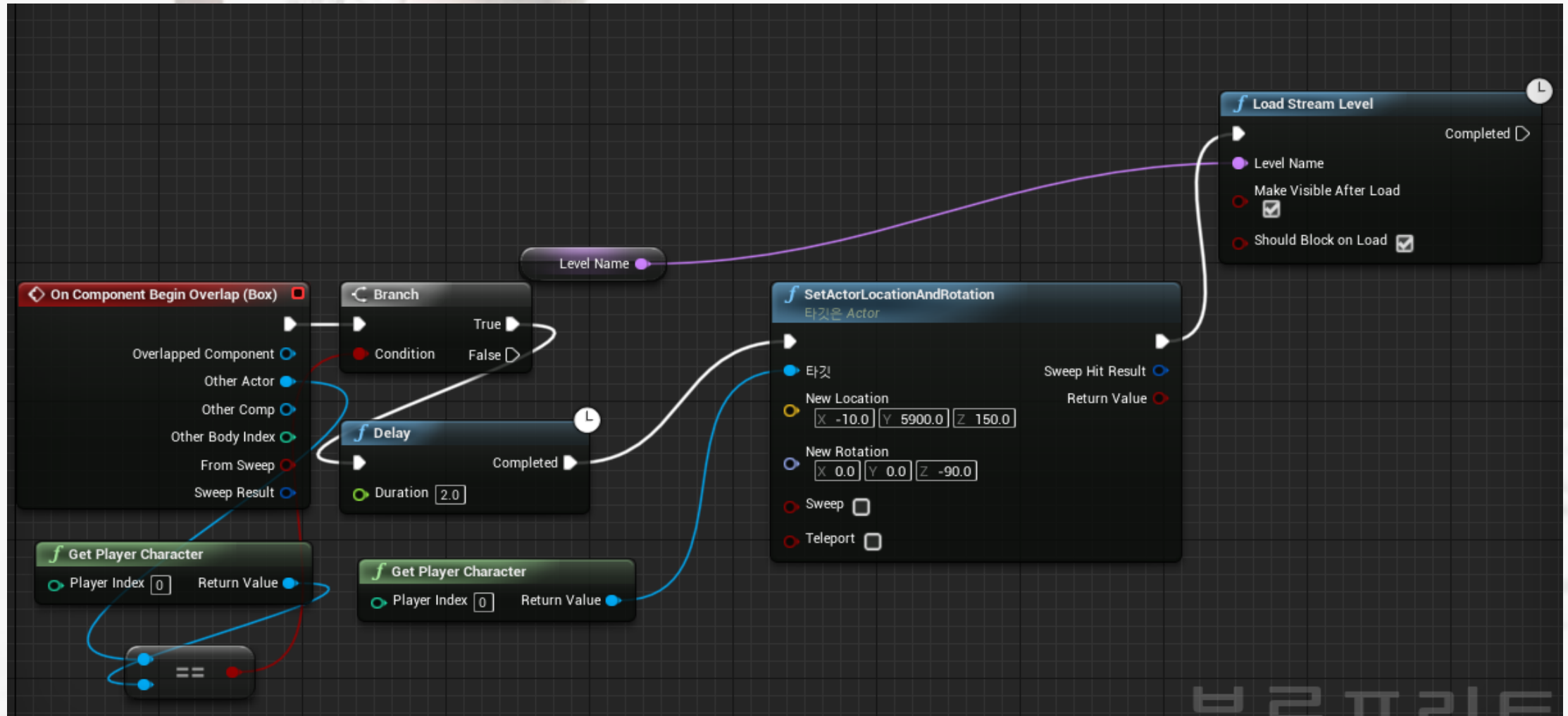

회피 기능



회피 기능



레벨 전환

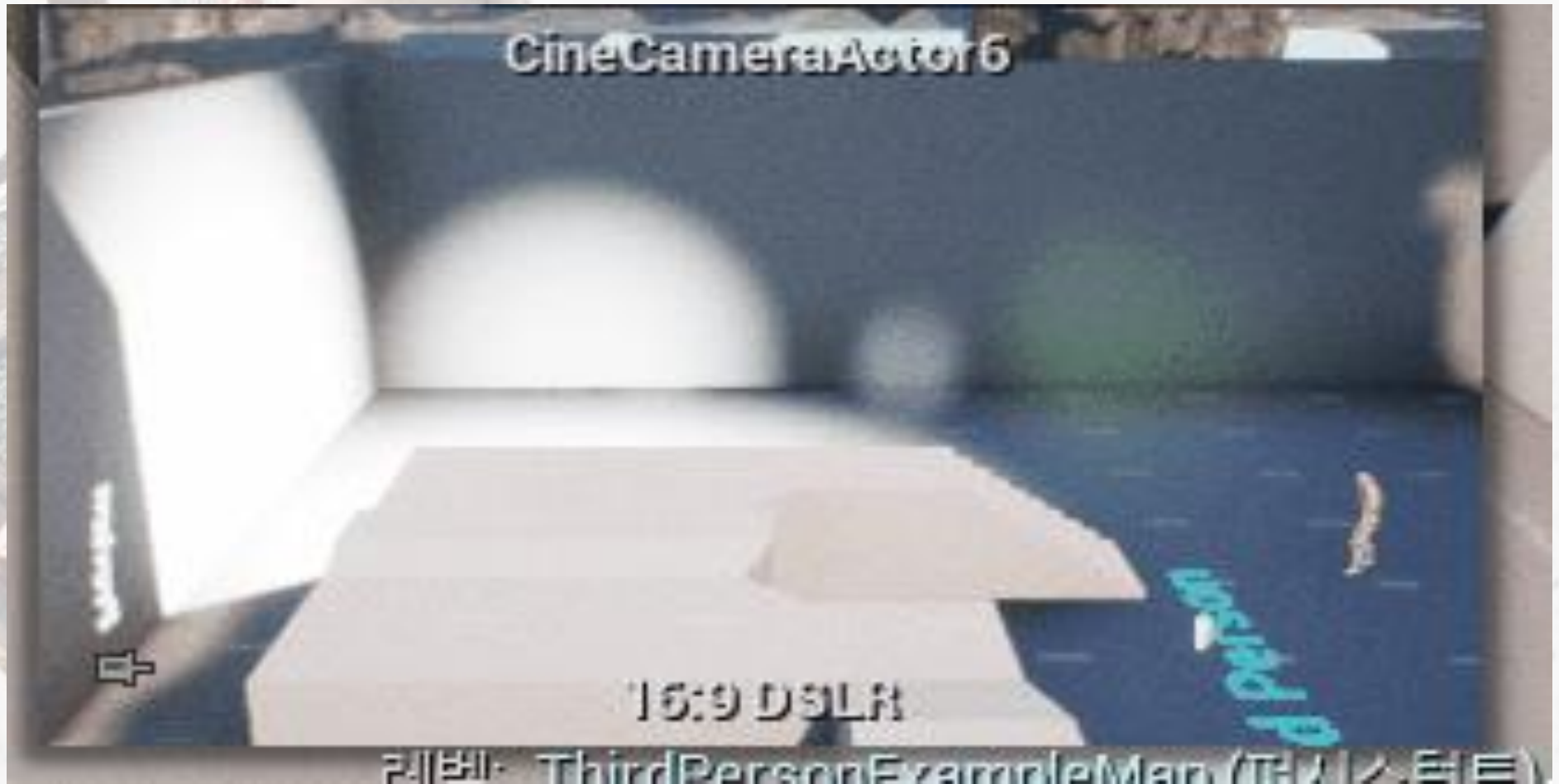


레벨 전환

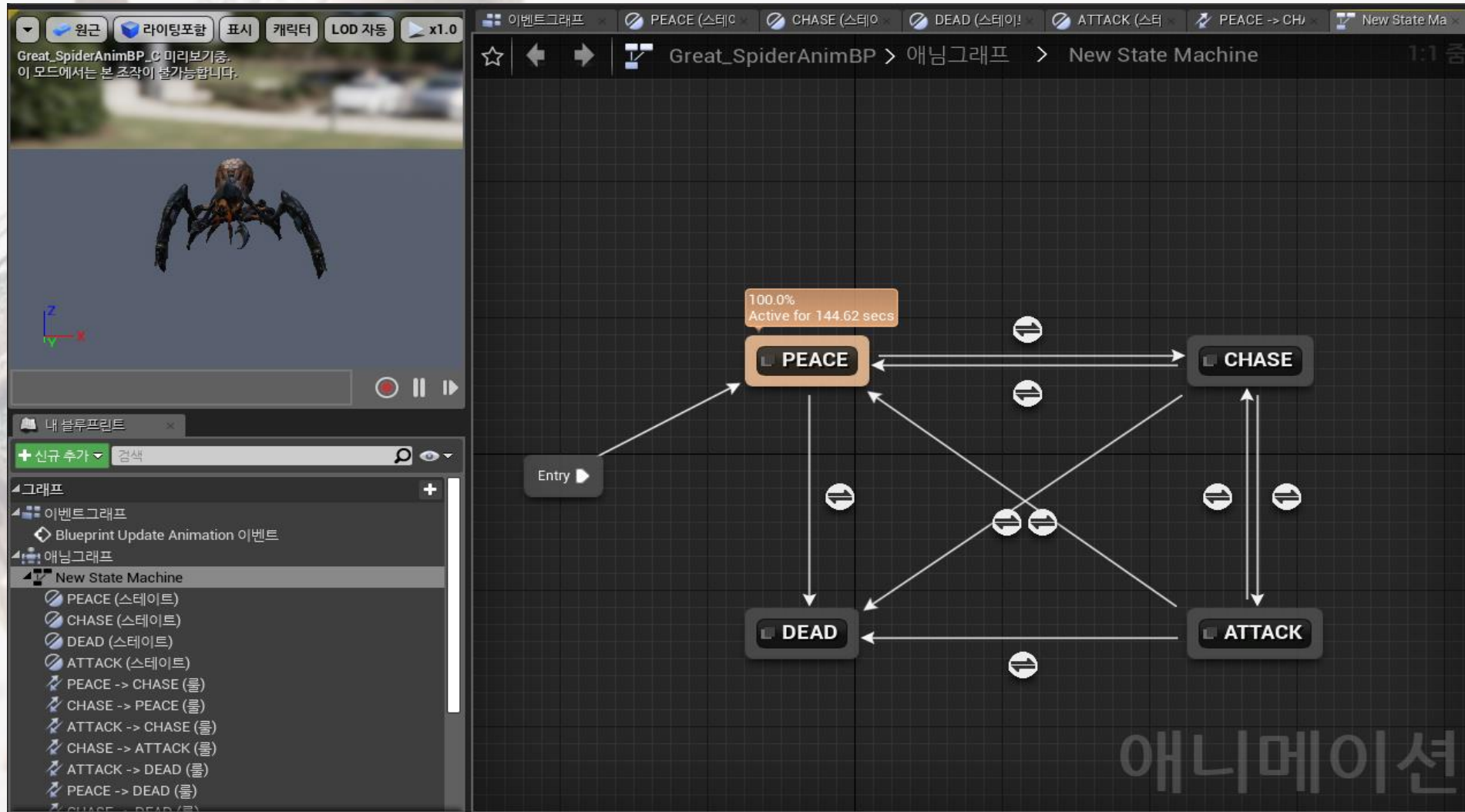


2018- ThirdPersonExampleMain (TH_12_31E)

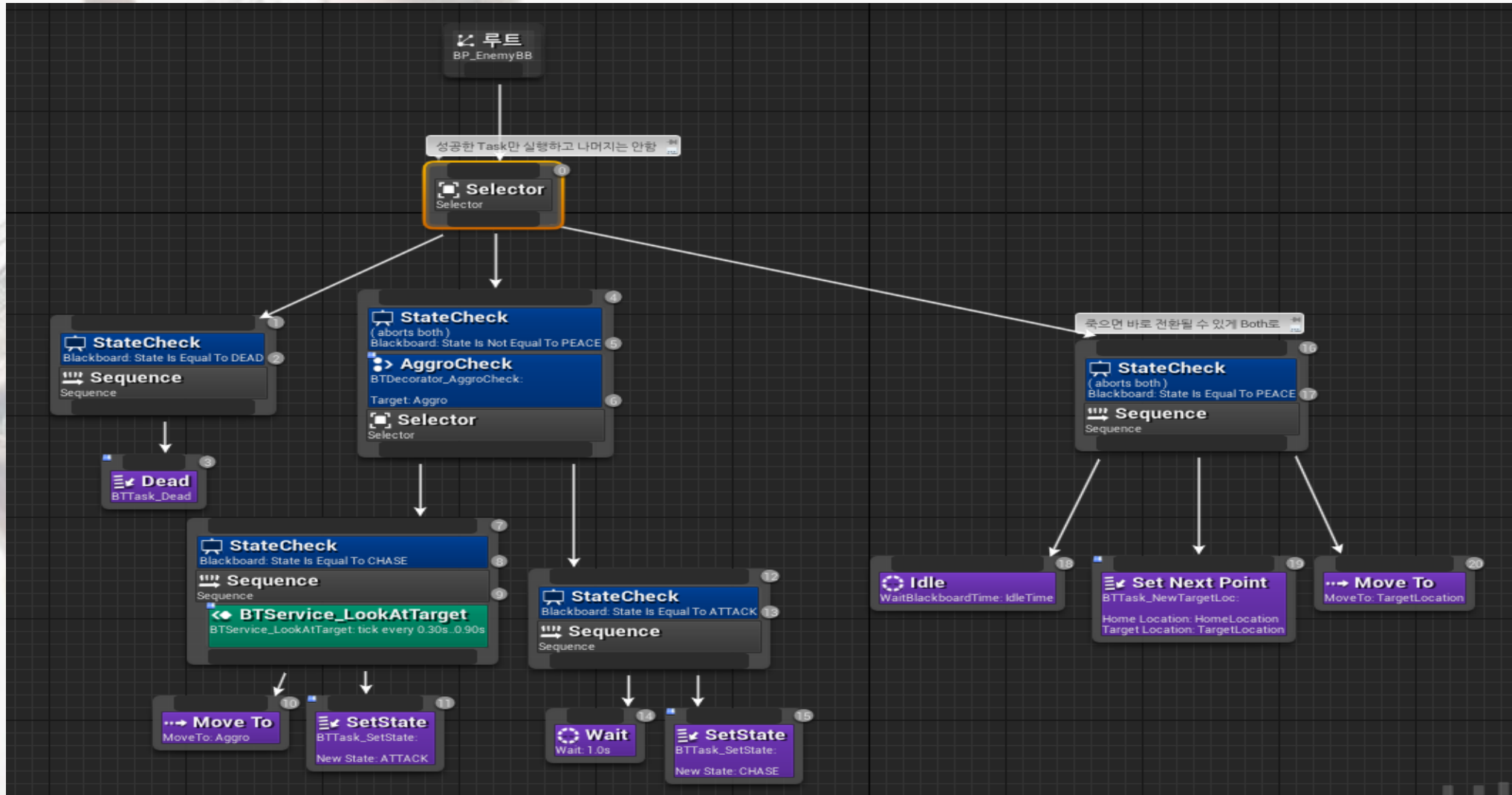
연출



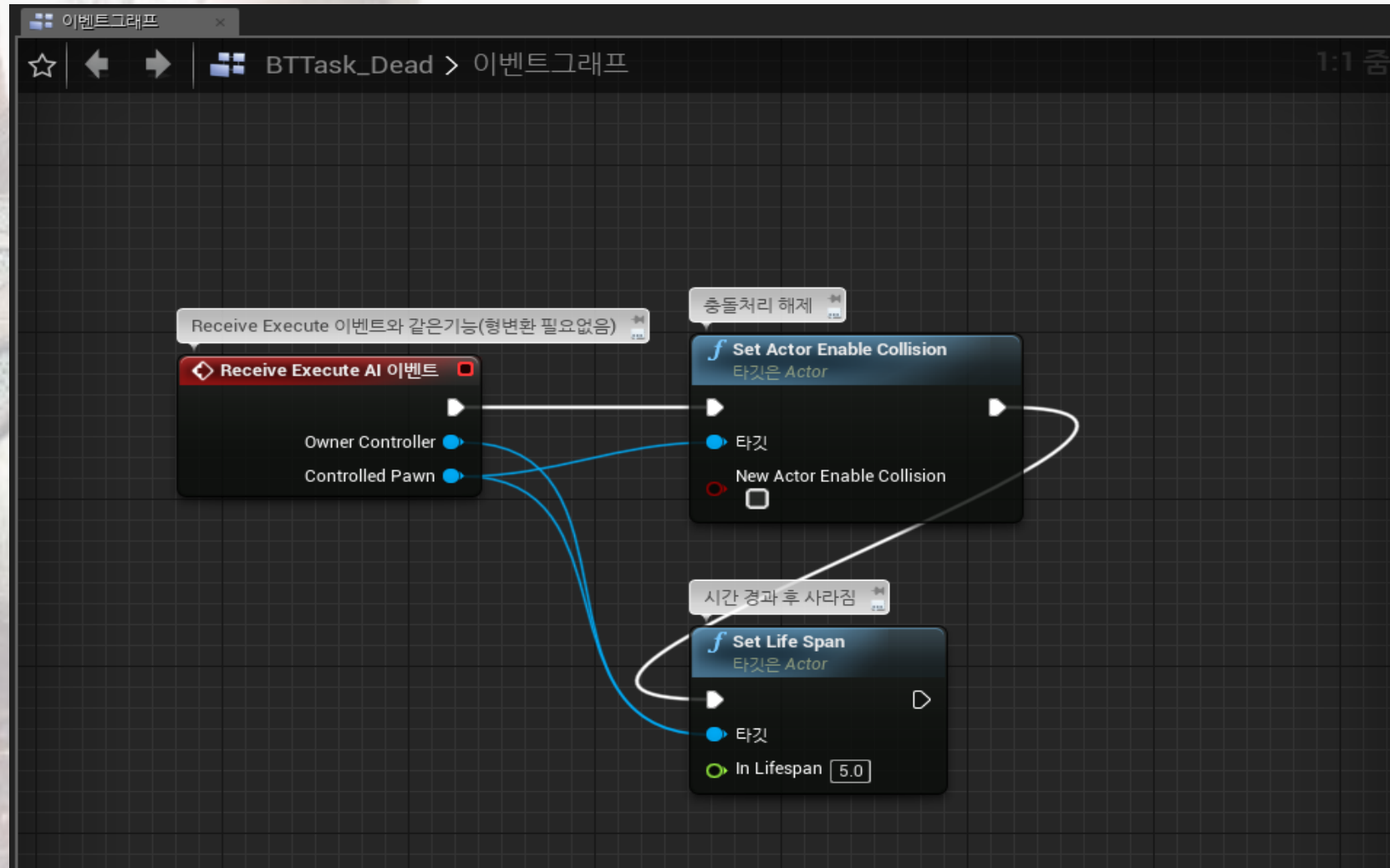
Monster 구현



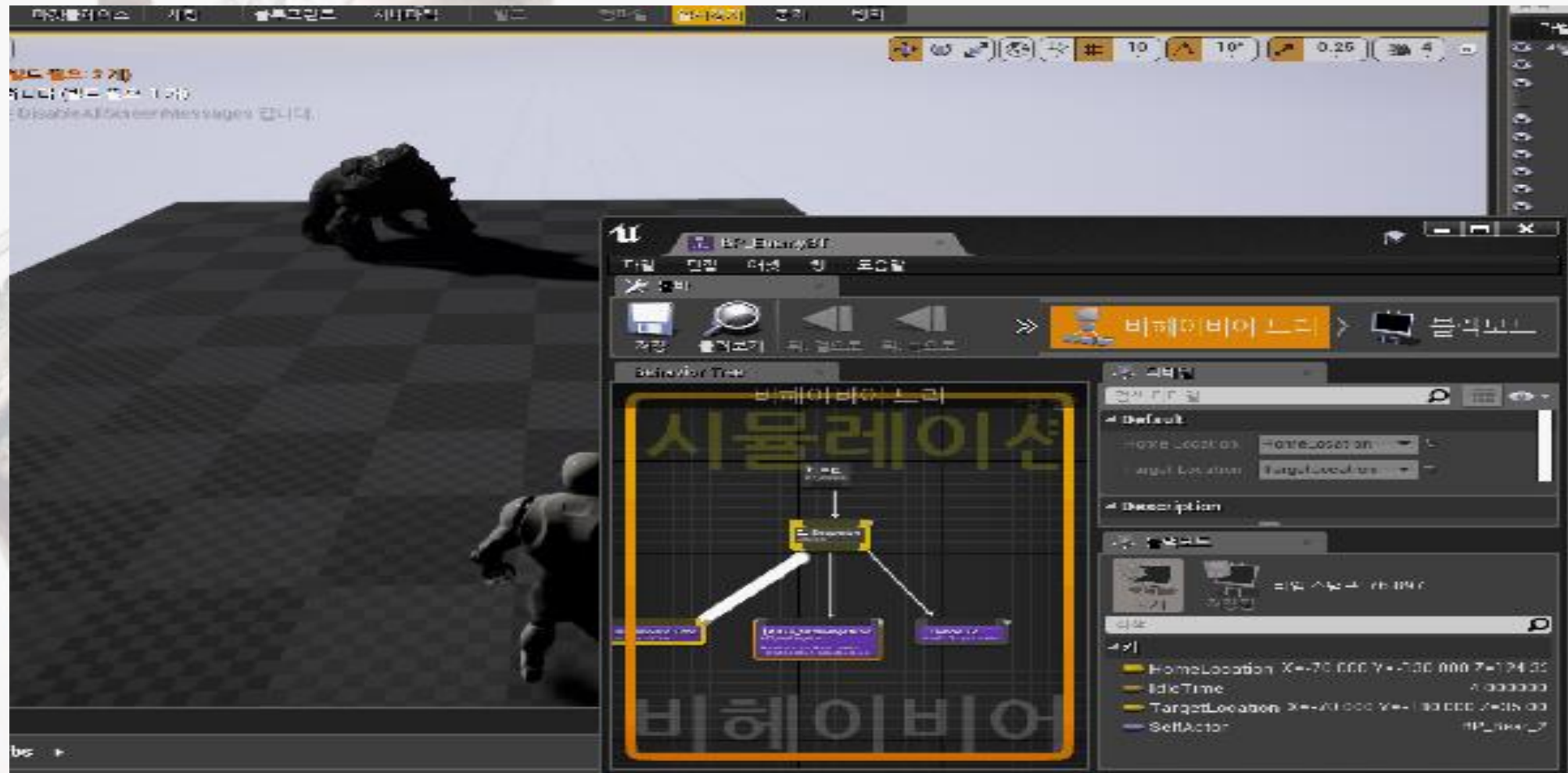
Behavior Tree



Behavior Tree

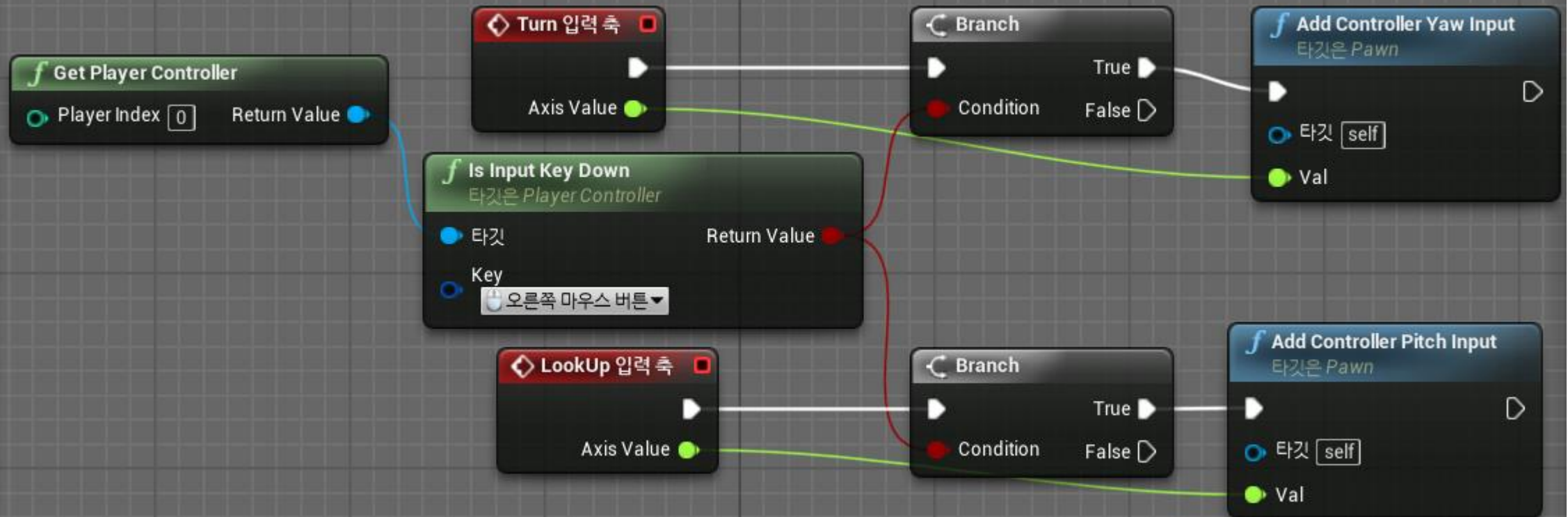


Behavior Tree



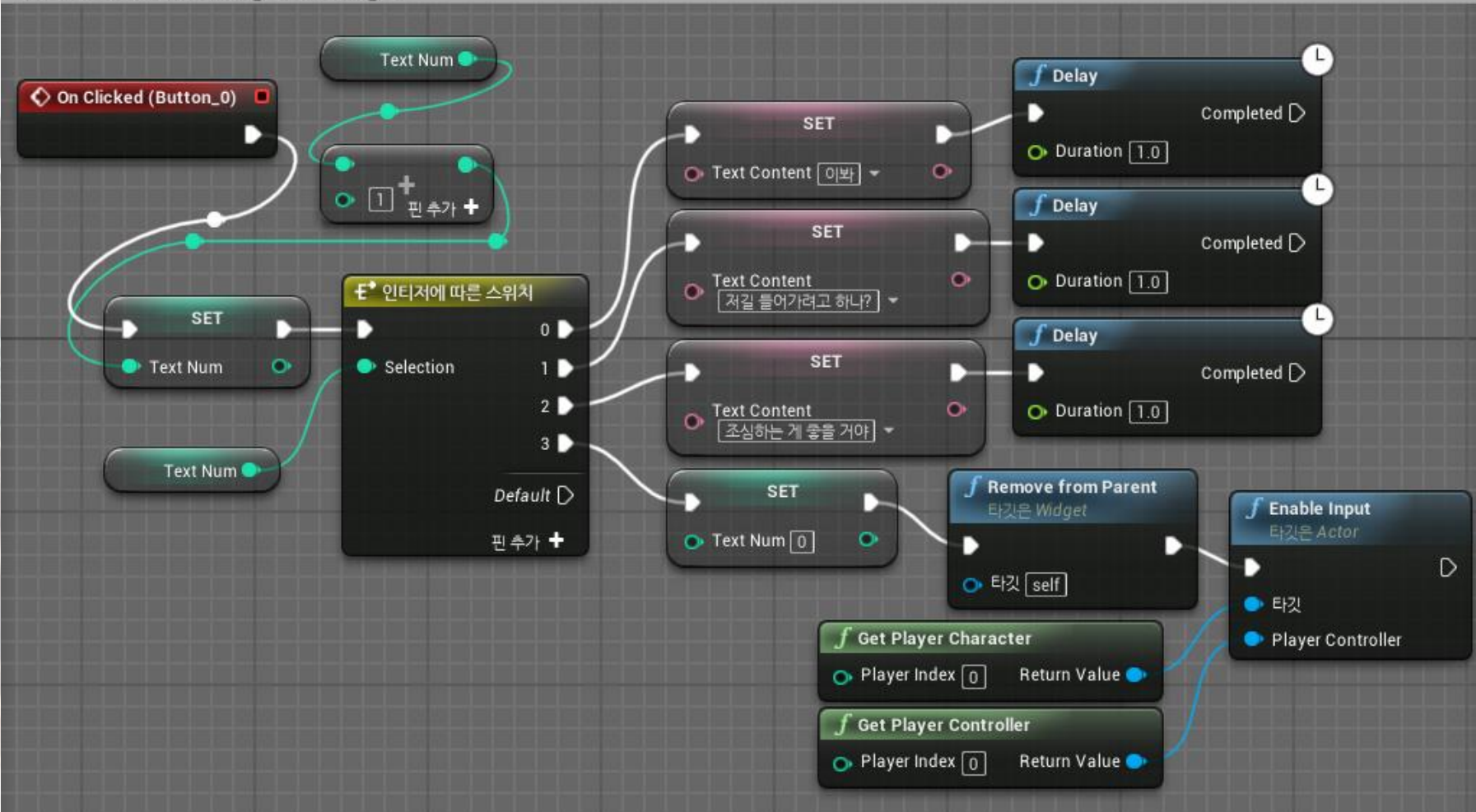
Mouse 시점 변환

Mouse rotation input

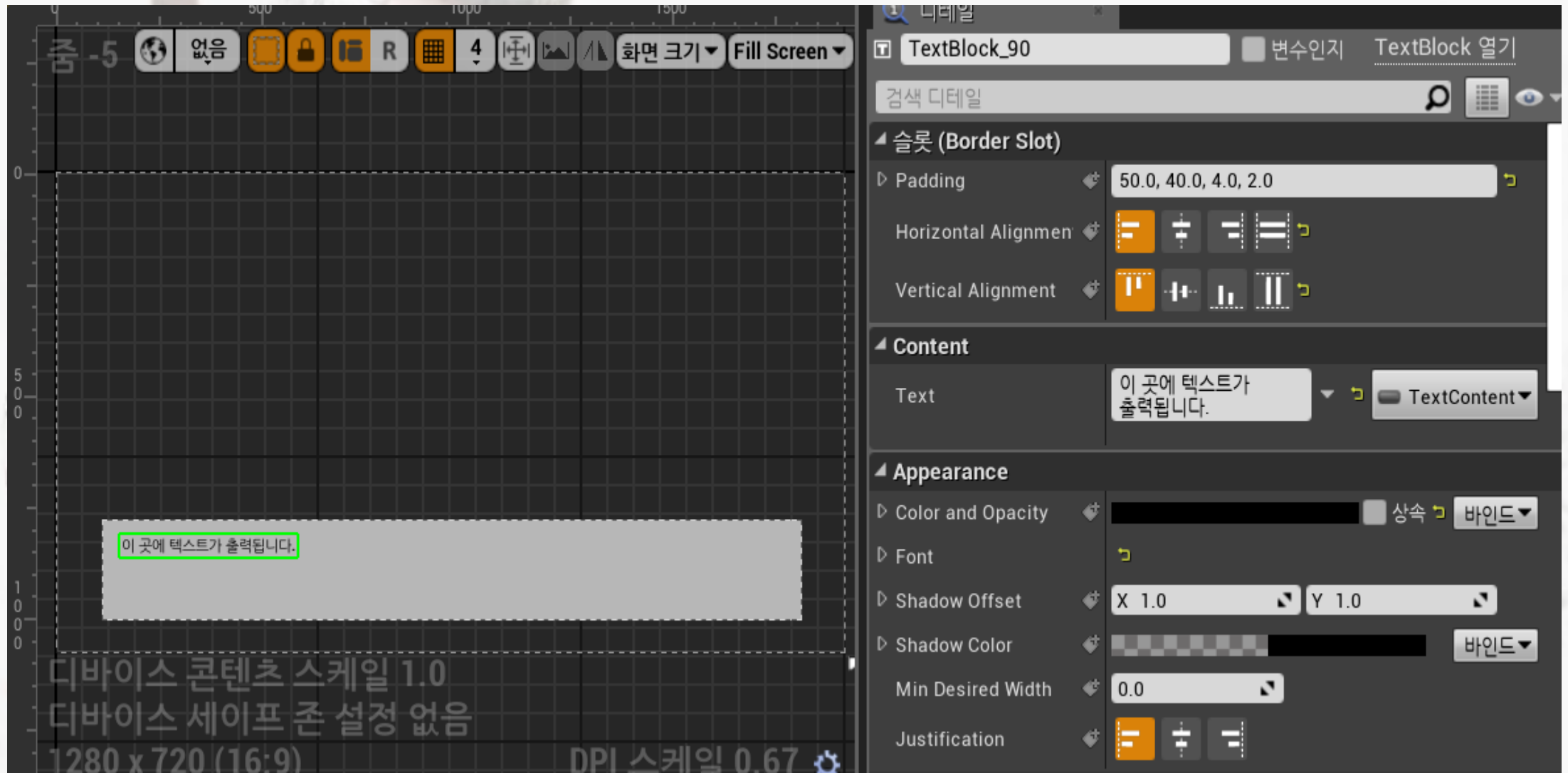


대화창 기능

Event for message change



대화창 기능



대화창 기능



시연 영상

<https://www.youtube.com/watch?v=ruiLo79qhc&feature=youtu.be>



향후 계획

항목	세부내용	12월	1월	2월	3월	4월	5월	비고
아이디어	아이디어 조사							
	기획 회의							
프로토타입 기획	엔리얼 엔진 튜토리얼							
	프로토타입 기획							
	리소스 조사							
프로토타입 개발	임시 플레이어 개발							
	플레이어 이동&회전							
	플레이어 회피 기능							
	NPC 대화창 구현							
	몬스터 개발							
	몬스터 추적 AI							
	맵 배치							
	맵 전환							
최종 기획	중간 피드백 및 최종 기획 회의							
개발	플레이어 움직임							
	전투 시스템							
	인벤토리 구현							
	아이템, 무기 제작							
	AI 시스템							
	각종 애니메이션							
	게임 화면 인터페이스							
	각종 UX / UI							
	퀘스트 시스템 구축							
	카메라 연출							
	데디케이티드 서버 구축							
	게임 내 DB 시스템 구축							
	로그인 / 환경설정							
	최종 게임 디버깅							
마무리	레벨 디자인 조정							

향후 계획

이름	개발항목	시작일	종료일	총개발일(MD)
전원	아이디어 조사 및 선정	2018-12-20	2018-12-31	5
전원	엔리얼 엔진 튜토리얼	2019-01-01	2019-02-28	7
전원	Prototype 시나리오 기획	2019-02-01	2019-03-31	5
전원	Prototype 구상 및 리소스 조사	2019-04-01	2019-04-19	3
김성훈	플레이어 기본 움직임	2019-04-01	2019-04-07	2
김종균	플레이어 회피 기능(스킬 추가)	2019-04-01	2019-04-10	2
장윤지	플레이어 움직임 및 마우스 커서 작업	2019-04-01	2019-04-08	3
송영륜	몬스터의 플레이어 추적 AI 기능	2019-04-01	2019-04-12	4
김성훈	플레이어 생성 및 카메라 회전	2019-04-08	2019-04-12	3
장윤지	NPC 대화 텍스트 작업	2019-04-08	2019-04-18	4
김종균	맵 설계 / 레벨 전환 / Fade In, Out	2019-04-10	2019-04-18	4
김성훈	플레이어 애니메이션 및 콤보 기능	2019-04-12	2019-04-18	3
송영륜	몬스터 설계(생성) 및 애니메이션 적용	2019-04-12	2019-04-18	3
전원	캡스톤 중간점검 피드백 및 기획 수정	2019-04-27	2019-05-05	5
김성훈	아이템 상자 / 무기 / 게임 데이터	2019-05-06	2019-05-12	3
김성훈	AI 컨트롤러와 비헤이비어 트리 연동	2019-05-13	2019-05-19	4
김성훈	게임 데이터 관리	2019-05-20	2019-05-26	2
김종균	콤보 관련 애니메이션, 인벤토리	2019-05-06	2019-05-12	5
김종균	퀘스트 시스템, 카메라 연출	2019-05-13	2019-05-19	3
김종균	데디케이티드 서버	2019-05-20	2019-05-26	2
장윤지	오브젝트 AI, 전투시스템	2019-05-06	2019-05-12	4
장윤지	각종 UI 위젯, 각종 인터페이스 시스템	2019-05-13	2019-05-19	5
장윤지	인터페이스 관련 종합 UI 작업	2019-05-20	2019-05-26	2
송영륜	플레이어 애니메이션, 전투 시스템	2019-05-06	2019-05-12	4
송영륜	종합 아이템 설계	2019-05-13	2019-05-19	3
송영륜	최종 레벨 디자인 작업	2019-05-20	2019-05-26	4
전원	전체 시스템 최종 디버깅 작업	2019-05-27	2019-05-31	5



Q/A