언리얼 기초

□ 기초 코드

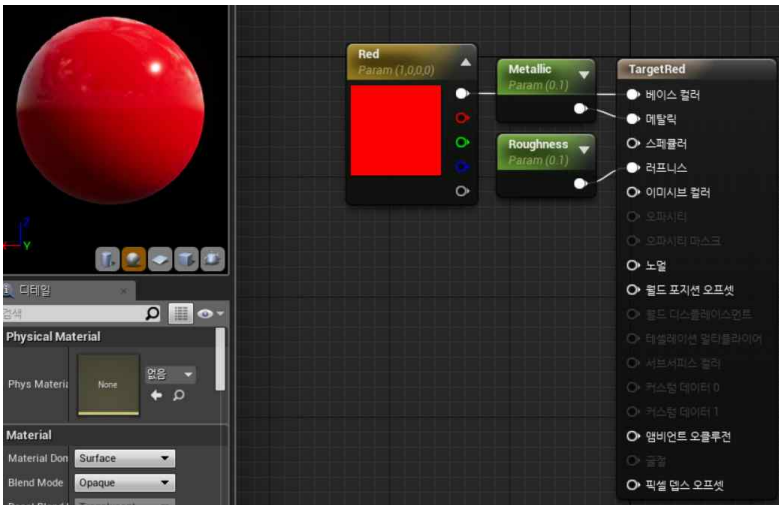
○ 디버깅 모드

* 디버깅 모드 :: 실행 - ` 입력 - ToggleDebugCamera 입력

□ 마테리얼

○ 마테리얼 생성

* 콘텐츠 브라우저 - Material 생성 - BP
* 베이스 컬러 설정 : VectorParameter 추가 - 색 설정 - 베이스 컬러(Base color)에 연결
* 반사(메탈릭) 설정 : ScalarParameter 추가 - Default Value 수치 설정(0.1~1.0) - 메탈릭(Metallic)에 연결
* 깊이(Roughness) 설정 : ScalarParameter 추가 - Default Value 수치 설정(0.1~1.0) - 러프니스(Roughness)에 연결

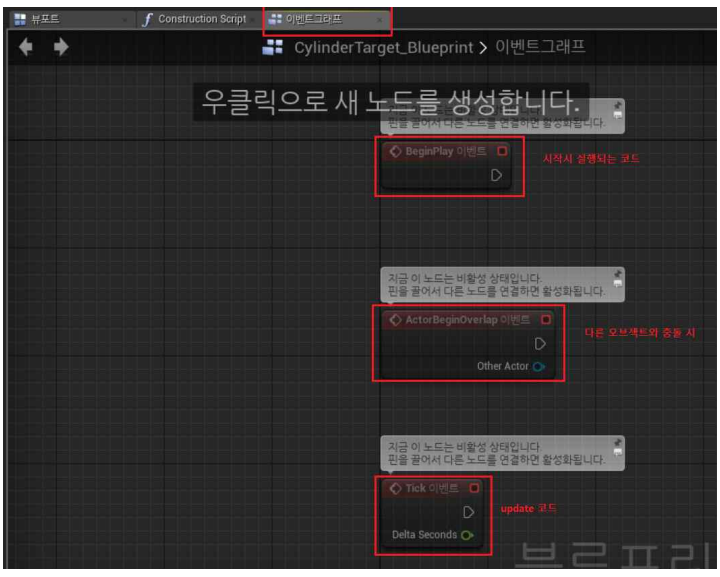


블루프린트

□ 오브젝트

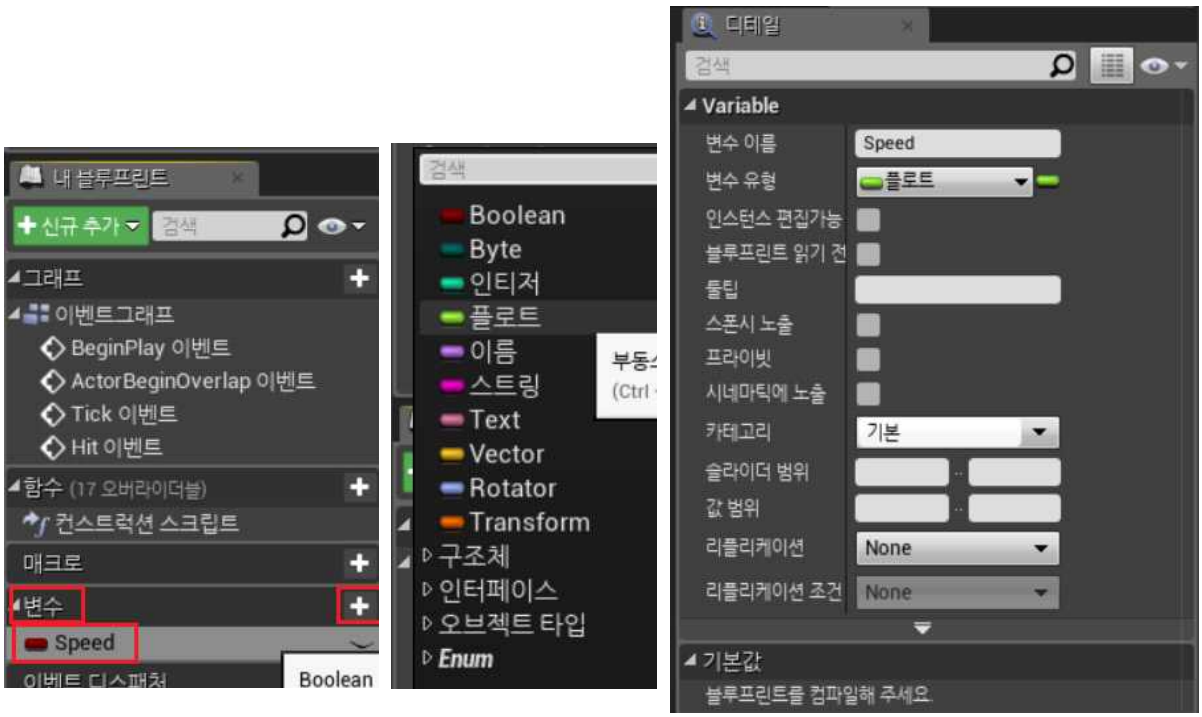
○ 블루프린트 추가

* 오브젝트 선택 - 우측 디테일 탭 - 블루프린트/스크립트 추가 - 이벤트 그래프 코드 추가
  - BeginPlay : 시작 시 실행되는 코드 (Create)
  - Actor Begin Overlap : 충돌 시 실행되는 코드 (Collision)
  - Event Trick : 계속해서 실행되는 코드 (Step)



○ 변수 값 저장

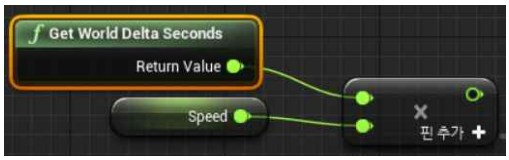
* BP – 내 블루프린트(My Blueprint) – 변수(Variables) – '+' 버튼으로 추가
  - 변수 이름(Variable Name)
  - 변수 타입(Variable Type)
  - 편집 가능(Editable) : 블루프린트 외부에서 값 수정 가능
  - 기본 값(Default Value) : 값을 수정할 수 있는 필드가 없고, 블루프린트를 컴파일 하라는 메시지만 있다 해당 변수의 초기 값을 지정한다.



* 변수 설정
  - Get : 변수의 값을 받아옴
  - Set : 변수의 값을 설정

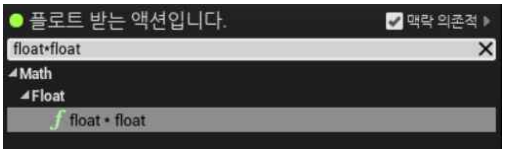
* 정규화(Normalize) : 벡터의 방향으로 표시

* 델타 타임 : 게임 플레이 프레임들 사이의 시간
  - Get World Delta Seconds : 델타 타임 반환



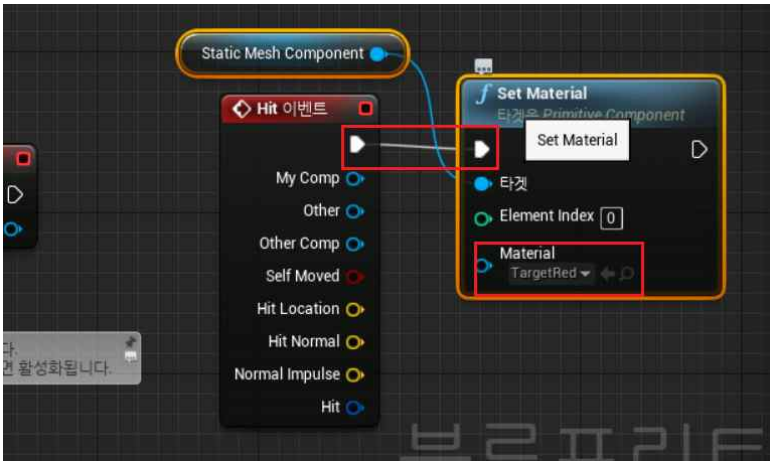
○ 연산자

* 빈 공간에 변수 타입 + (연산자) + 변수타입를 입력 하여 추가

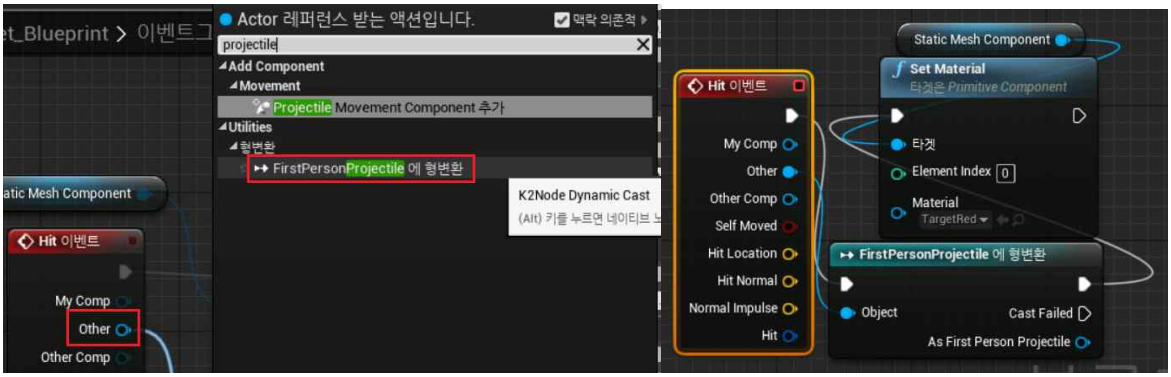


○ 오브젝트 충돌 시 마테리얼 변경

* Event Hit 추가 - Set Material(StaticMeshComponent) 추가 :: Material -> 생성되어있는 마테리얼 설정



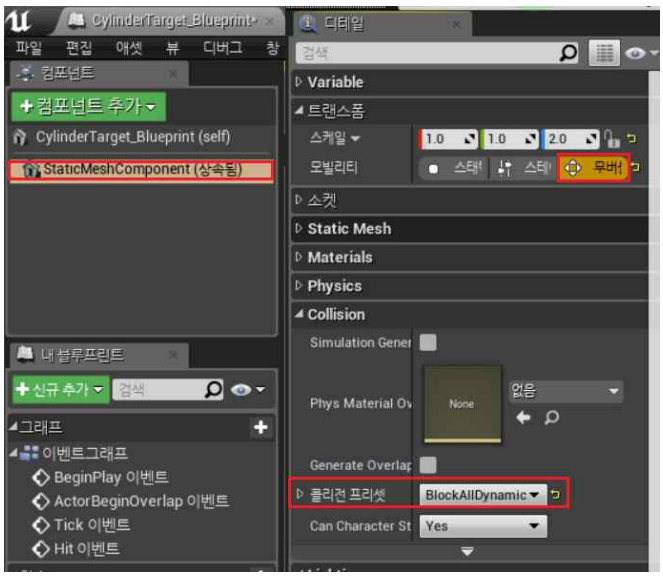
* FirstPersonProjectile에 형변환 설정 - Hit 이벤트와 Set Material 사이를 연결



○ 오브젝트 이동 / 충돌

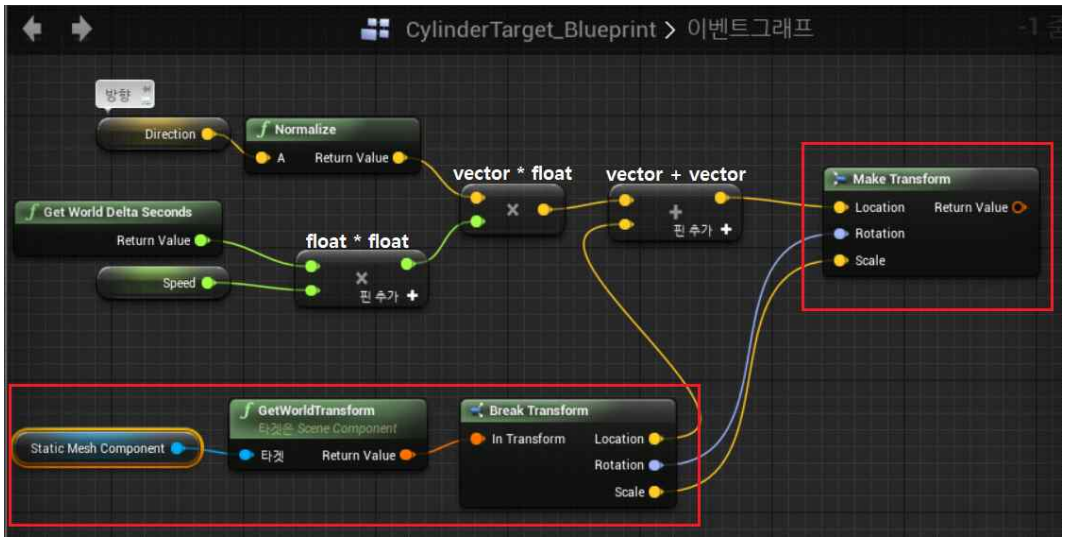
* 디테일 - 트랜스폼 - 모빌리티 - 무버블 설정
  - 스태틱 오브젝트 : 게임에서 변경 불가(고정), 라이팅 굽기 가능, 렌더링 가장 빠름
  - 스테이셔너리 오브젝트 : 게임에서 변경 가능, 움직이지 않으며, 캐시 방식의 라이팅 메소드 사용가능
  - 무버블 오브젝트 : 동적이며, 다이나믹 섀도우 사용, 렌더링 가장 느림

* BP 안의 컴포넌트 - StaticMeshComponent - 디테일
  - 트랜스폼 - 모빌리티 - 무버블 설정
  - Collision - 콜리전 프리셋 - BlockAllDynamic 설정
  - BlockAllDynamic : 타겟 오브젝트가 모든 오브젝트와 충돌 반응이 가능하도록 설정



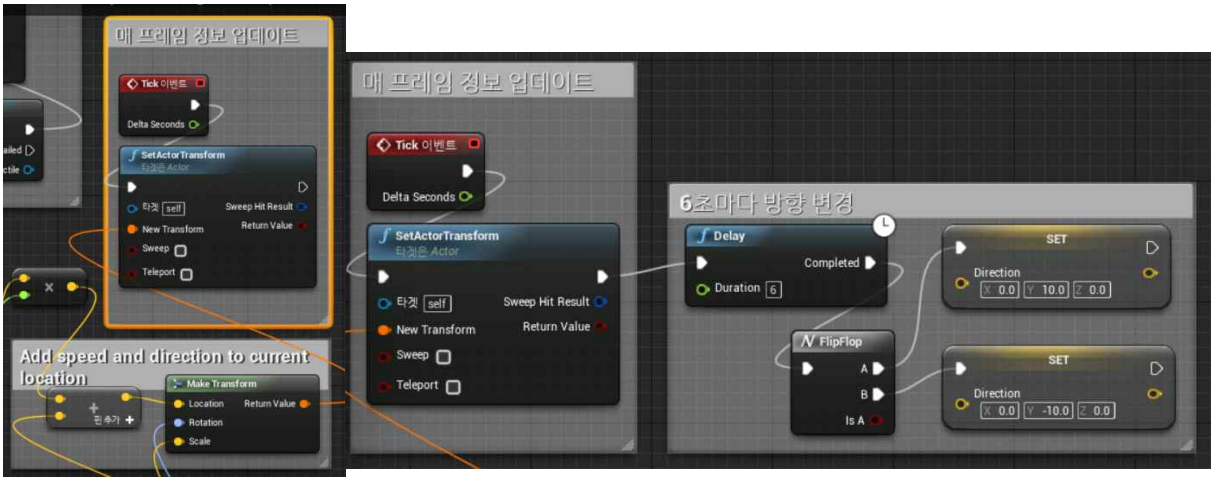
○ 오브젝트 전역 변수 받아오기

* StaticMeshComponent : 오브젝트 메시 컴포넌트가 갖고 있는 모든 데이터를 담고 있다
  - Get World Transform : 오브젝트의 회전, 스케일, 위치 값 반환
  - Break Transform : World Transform을 분해하여 필요한 변수로 나눌 수 있다
  - Make Transform : 새로운 변환을 만든다



○ 위치 값 업데이트 및 방향 바꾸기

* Event Tick : 델타타임을 지속적으로 받아옴
* Set Actor Transform : 새로운 트랜스폼 정보를 설정
* FlipFlop : 두 개의 변수를 번갈아 가며 실행하는 함수
* Delay : 시간을 지연시킬 수 있는 함수 (알람)

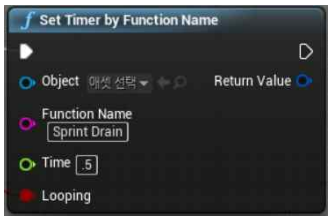


○ 참 / 거짓

* Branch : 참 / 거짓의 결과를 출력
  - Condition : 비교한 Bool 값 받아옴

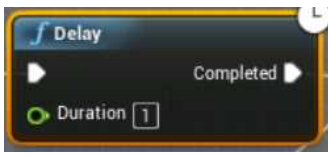
○ 재귀함수 :: 딜레이(알람)

* Set Timer by Function Name : 해당 함수에 대하여 알람기능 적용
  - Object : 대상 오브젝트
  - Function Name : 실행할 함수 이름
  - Time : 몇 초마다 실행할 것이지 설정
  - Looping : 반복할 것인지 설정



* Clear Timer by Function Name : 해당 함수에 대하여 알람기능 해제
  - Object : 대상 오브젝트
  - Function Name : 해제할 함수 이름

* Delay : 딜레이를 적용
  - Duration : 딜레이 시간 설정



○ if 문 => Branch

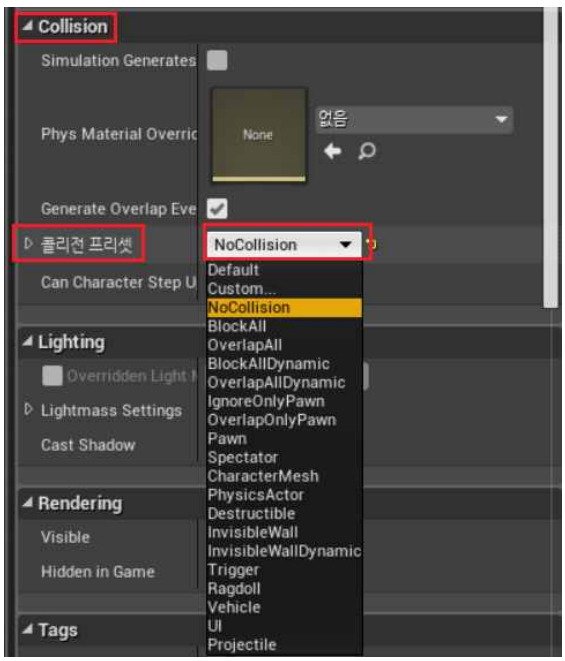
○ int * int 형식으로 연산자 표시

○ 멀티탭 기능 - Sequence

* 여러 개의 노드들을 연결할 수 있다.

○ 오브젝트 충돌 감지 - Collision

* NoCollision : 충돌되어도 막히지 않음(기체화)
* Block All : 충돌되면 막힘(고체화)

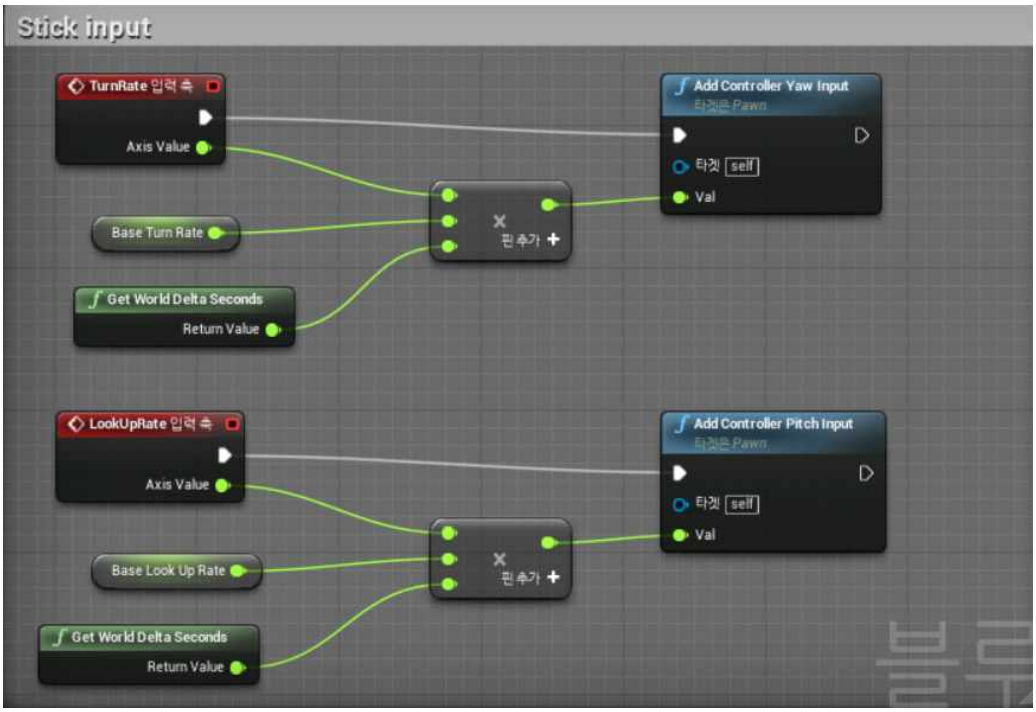


□ 플레이어

○ 플레이어 입력 (Stick input)

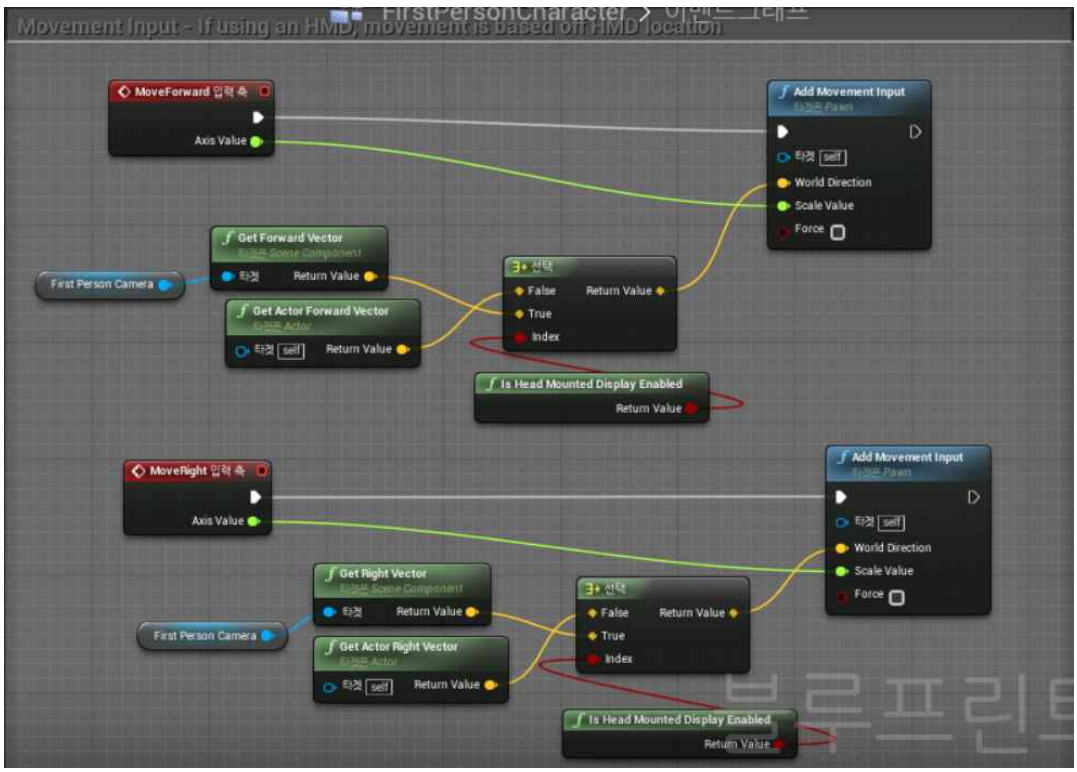
* Base Trun Rate : 플레이어 회전 속도 결정
* Base Look Up Rate : 플레이어 위, 아래 바라볼 때 속도 결정
* 델타 시간과 곱해짐
* Add Controller Pitch Input : 컨트롤러 입력에 이동을 적용
* Add Controller Yaw Input : 플레이어 카메라에 이동을 적용

* Mouse Input : 마우스 입력 장치로부터 값을 얻어 카메라 요(Yaw)와 피치(Pitch)의 입력 함수로 전달



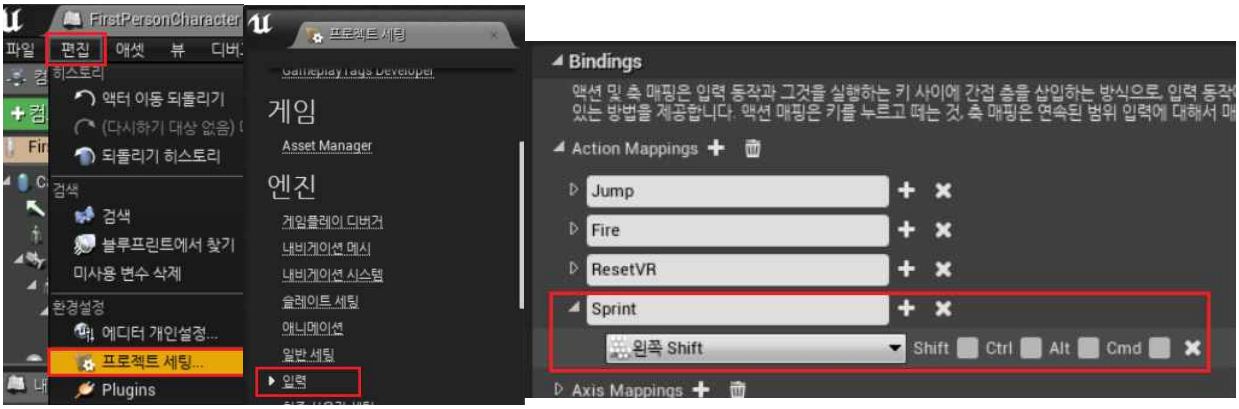
○ 플레이어 이동

* Get Actor Vector : 액터의 방향 반환
  - Add Movement Input의 World Direction에 연결



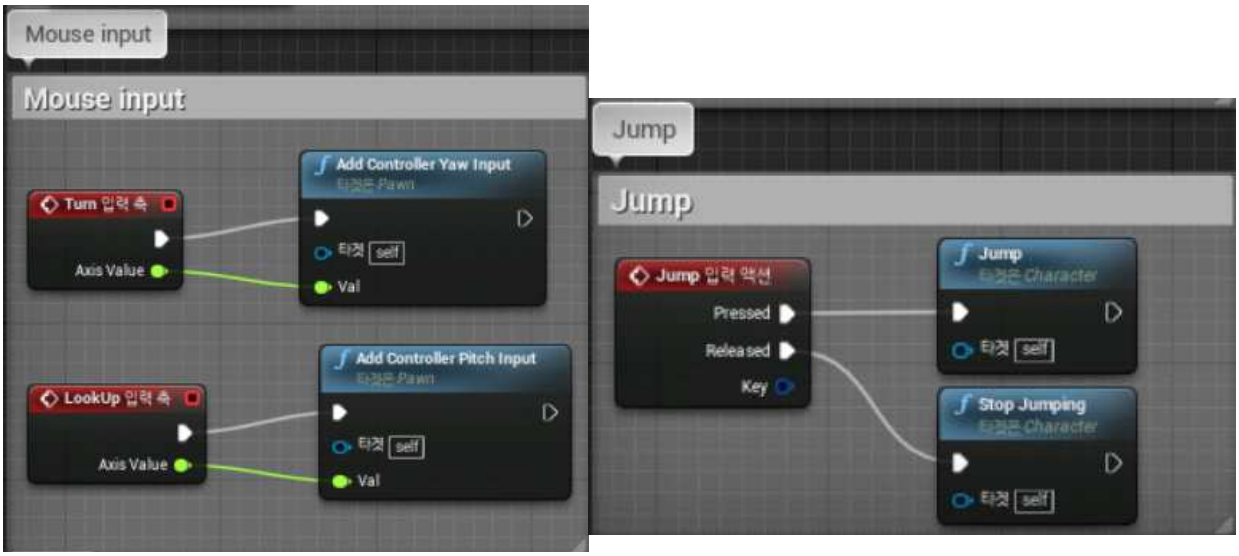
□ 입력 키 변경

* BP - 편집 - 프로젝트 세팅 - 엔진 - 입력 - Bindings
  - Action Mappings : 키보드 입력 / 마우스 클릭으로 플레이어의 액션 트리거 용도로 사용
  - Axis Mapptings : 플레이어 움직임을 매핑하고 범위를 가진다



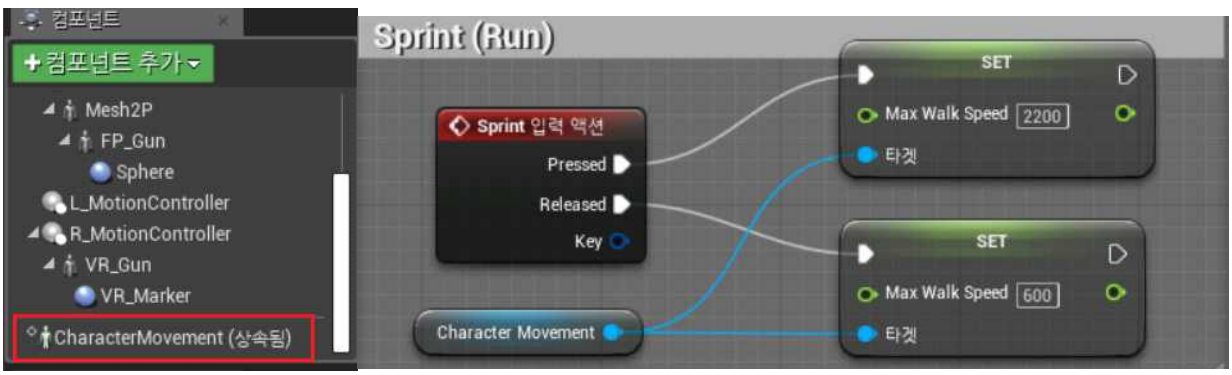
○ 플레이어 마우스 입력 및 점프

* 마우스 입력 및 점프



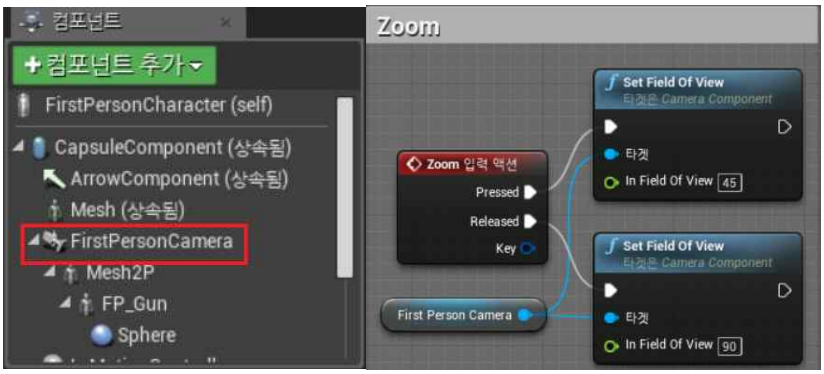
○ 플레이어 달리기

* Sprint : 입력 키에 등록했던 Sprint를 불러온다.
* Character Movement : 캐릭터 움직임에 관한 모든 기능을 가지고 있다
* Get Max Walk Speed : 플레이어의 최대 달리기 속도 반환
* Set Max Walk Speed : 플레이어의 최대 달리기 속도 설정

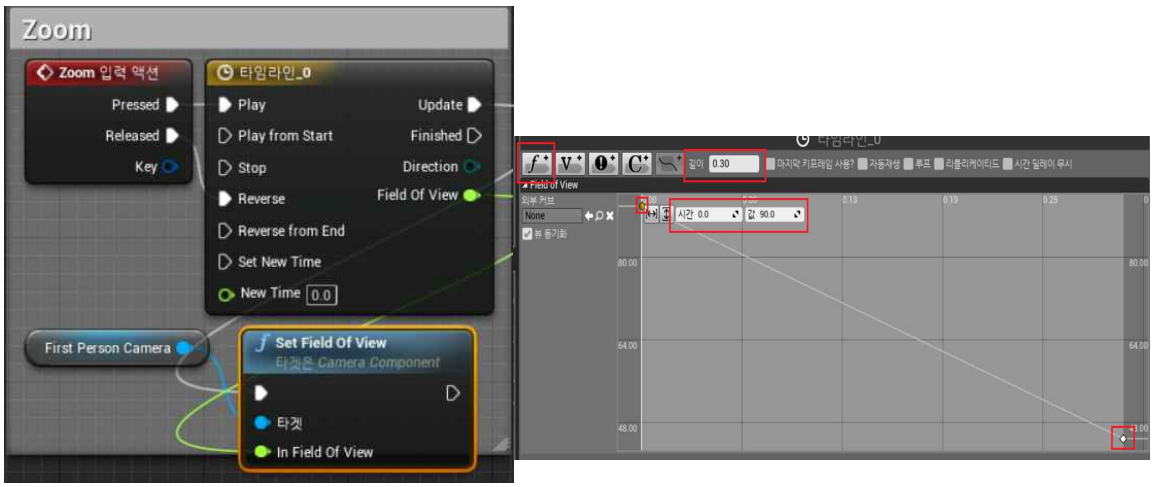


○ 플레이어 줌 기능

* FirstPersonCamera : 플레이어의 카메라의 모든 기능을 가지고 있다
* Set Field Of View : 줌 기능을 해줄 카메라의 기능을 설정한다. // 렌즈 각도를 좁히면 대상이 확대

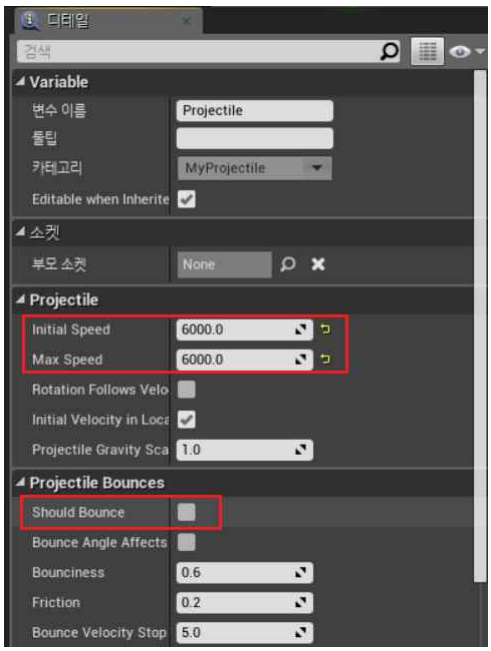


* 타임라인 : 일정 시간동안 값을 변화하게 해준다 (ex) 카메라의 FOV 값)
  - 위 보다 좀 더 부드럽게 줌이 됨
  - 0.3 초 간격으로 타임라인이 설정되어 있음



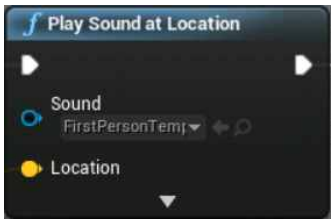
○ 총알 변경

* BP - Components 패널 - Projectile : 발사체의 모양과 충돌, 어떻게 움직이는지 정의
  - Initial Speed : 초기 속도
  - Max Speed : 최대 속도
  - Should Bounce : 바운스(튕기기) 유무 설정



□ 사운드, 파티클 이펙트

○ 사운드



* Play Sound at Location : 사운드를 재생
  - Sound : 사운드 파일 선택
  - Location : 사운드가 재생될 위치 설정

○ 파티클 이펙트

* 이미터(emitter) : 특정 위치에서 파티클 이펙트를 만들어내는 오브젝트
  - 파티클 이펙트는 하나 이상의 작은 오브젝트로 이루어져 있으며 시각 효과를 만들어 냄
  - 유체, 기체 등의 효과를 만듦 // 폭포수, 폭발, 라이트 빔 등

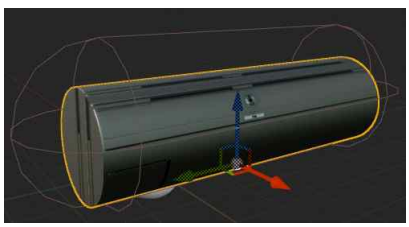
* Spawn Emitter at Location : 파티클 이펙트를 생성
  - Emitter Template : 이펙트 설정
  - Location : 위치
  - Rotation : 회전
  - Auto Destroy : 자동 소멸

○ 오브젝트

* Destroy Actor : 인스턴스 소멸

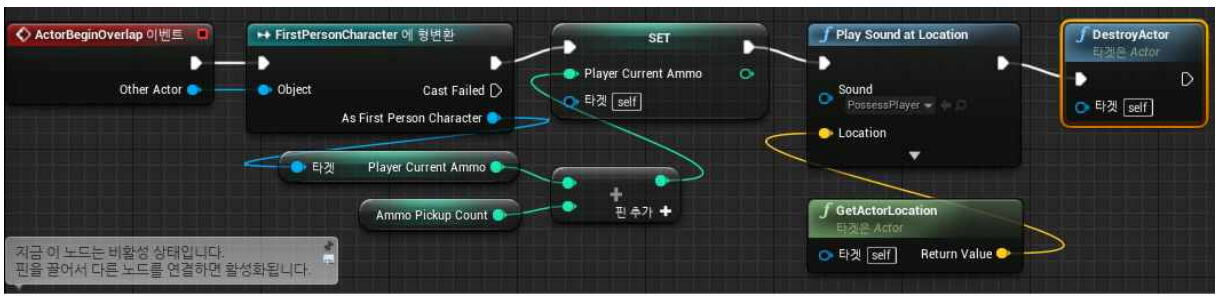
○ 충돌 범위

* BP - 컴포넌트 - 컴포넌트 추가 - capsule Collision



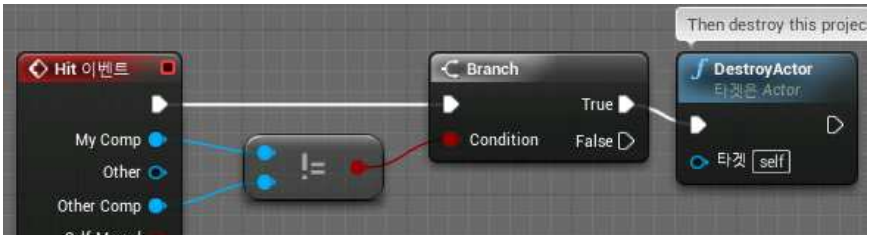
○ Casting(형변환)

* 오브젝트에서 다른 오브젝트의 변수를 받아올 때 사용
  - 형변환의 파란색 점으로부터 변수를 받아올 수 있다(. (점) 연산자)



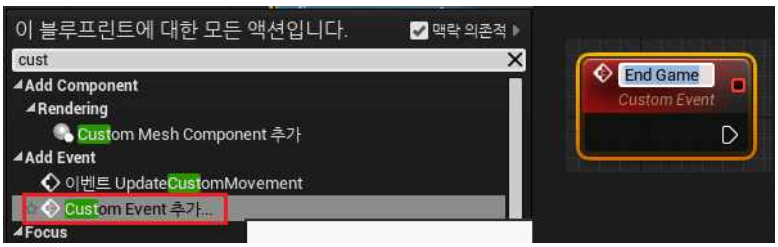
○ 오브젝트 비교

* NotEqual (Object) : 충돌 오브젝트가 해당 오브젝트와 같지 않은지 비교



○ 이벤트 함수 추가

* Custom Event 추가 : 이벤트 함수를 추가한다



□ 게임 상태

○ 게임 재시작

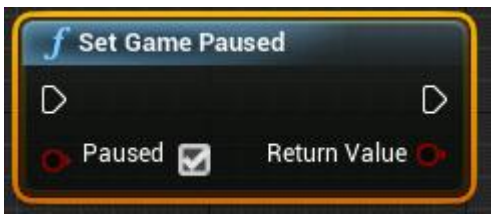
* Open Level : 레벨(룸)을 새로 시작한다.
* Remove from Parent : 불러온 함수의 오브젝트를 제거한다. (UI 제거)

○ 게임 종료

* Quit Game : 게임을 종료한다.

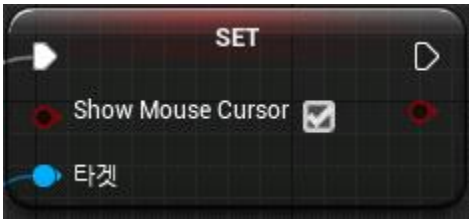
○ 게임 일시정지

* Set Game Paused : 게임 일시정지
  - Paused 체크시 일시정지



○ 마우스 커서 보이기

* Show Mouse Cursor : 마우스 커서를 보이게 한다



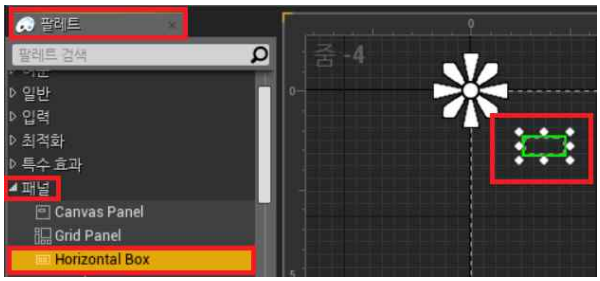
□ UI - UMG

○ UMG 생성

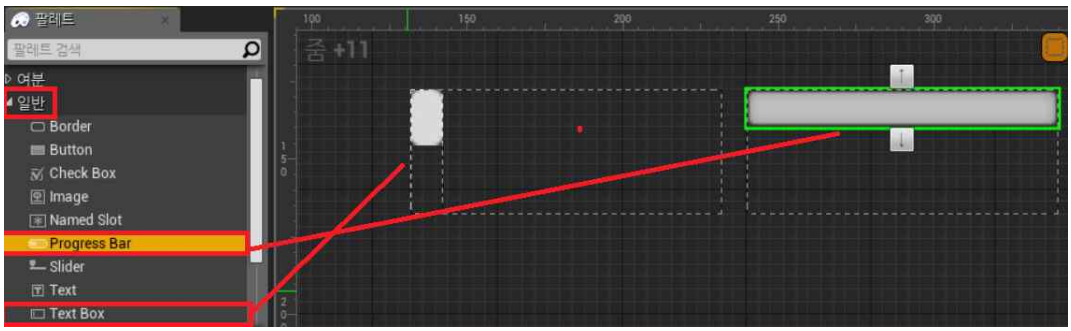
* 위젯 블루프린트 생성 - 더블클릭하여 UMG 편집기 열기
* 팔레트를 통하여 원하는 위젯을 만들 수 있다

○ 팔레트

* 패널
  - Horizontal Box(수평 박스)

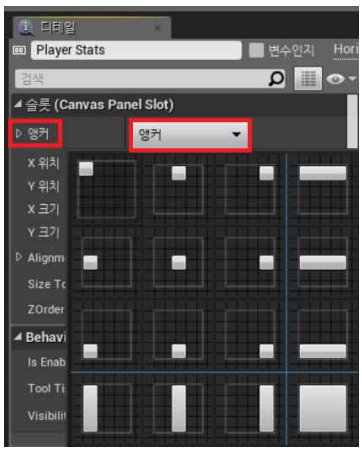


* 일반(Common)
  - Progress Bar
  - Text Bar

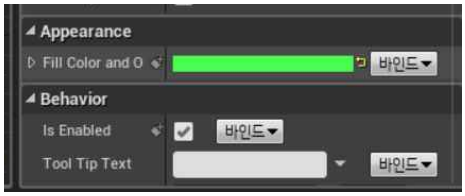


○ 디테일

* Horizontal Box(수평 박스)
  - Anchor : 윈도우 크기가 바뀔 때, 어느 위치에 붙을 것인지 설정

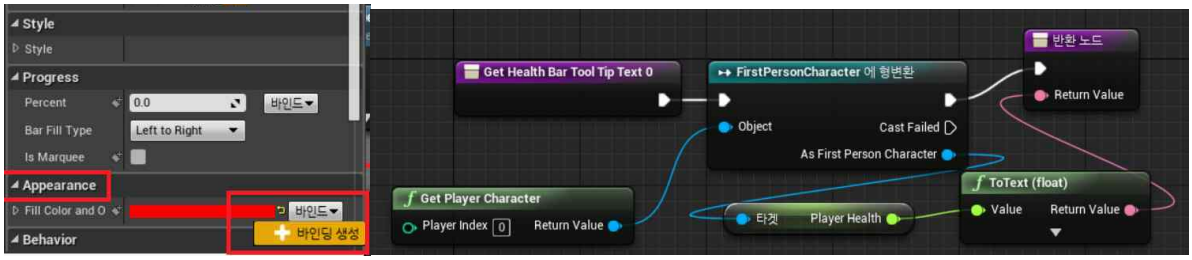


* 수평 박스 이외
  - Appearance :: Fill Color and Opacity : 색과 불투명도 설정

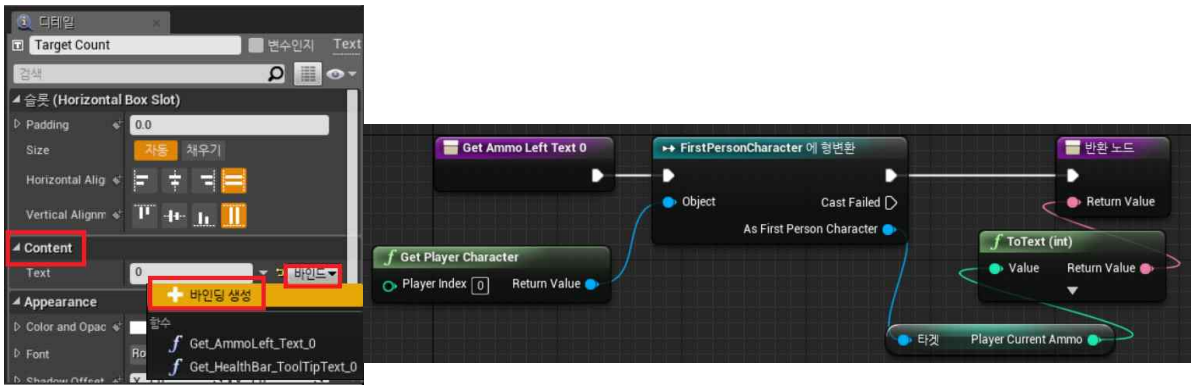


○ 바인딩

* 이미지 바인딩
  - 디테일 - Appearance :: Bind(바인드) - 바인딩 생성 : 연결해주는 바인딩을 생성한다.
  - Get Player Charactor : 플레이어의 정보를 받아옴
  - FirstPersonCharacter에 형변환 : FirstPersonCharacter로 형 변환
  - ToText(변수 타입) : 해당 변수 타입 값을 텍스트로 변환

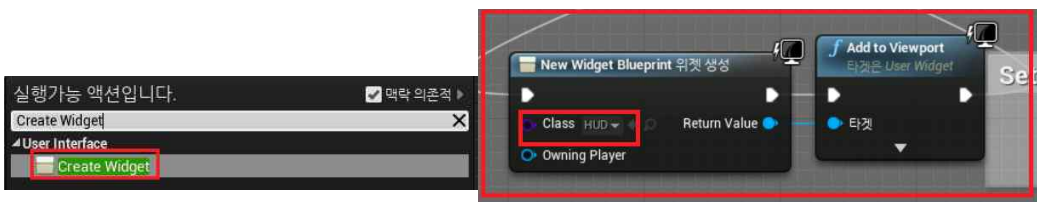


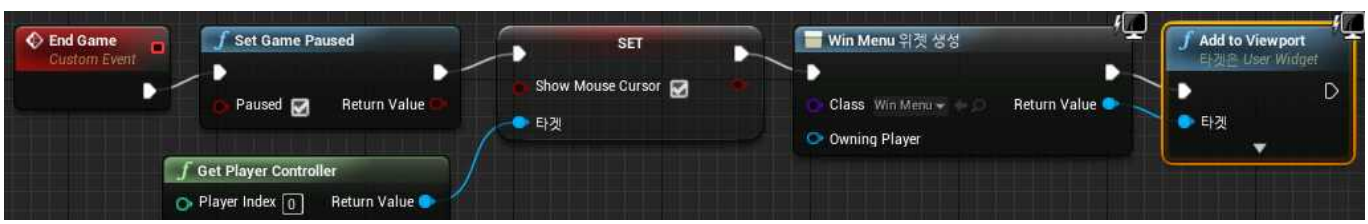
* 텍스트 바인딩
  - Content :: Bind(바인드) - 바인딩 생성 : 연결해주는 바인딩을 생성한다.



○ 위젯 추가

* 플레이어 BP에서 클래스 추가
  - CreateWidget : 위젯 생성 함수
  - Add to Viewport : 뷰 포트 추가





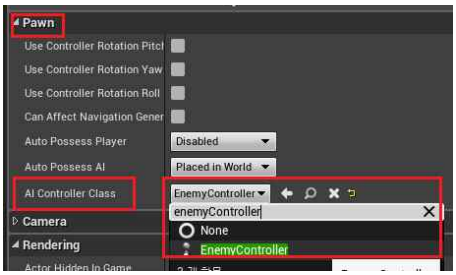
□ AI

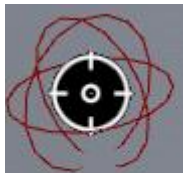
○ Nav Mesh 생성

* 모드 - 볼륨 - Nav Mesh Bound Volume : AI가 지나다닐 수 있는 길을 자동 지정
  - 스케일을 늘리면 자동으로 지정이 된다.
  - 'P'를 누르면 네비의 범위(연두색)가 보인다

○ Pawn BP Class

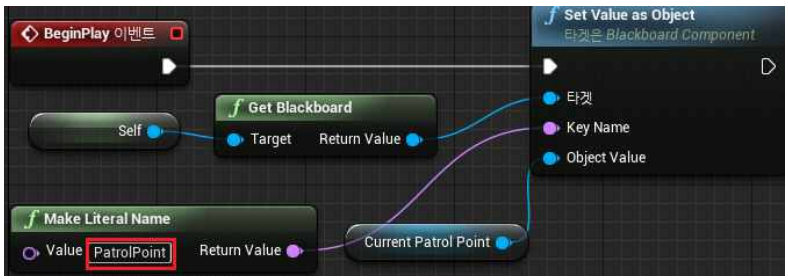
* 컨트롤러에 입력 받을 수 있는 오브젝트 BP
  - AI를 컨트롤하기 위해 디테일 - Pawn - AI Controller Class를 설정



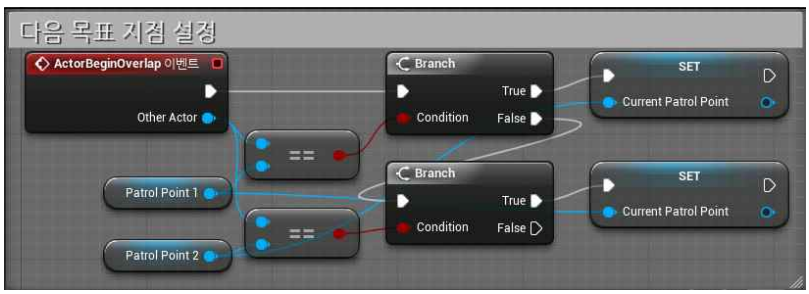


  - Actor 변수 PatrolPoint1 추가

* 첫 목표 지점 설정
  - Get Reference to Self(자기 레퍼런스 구하기) : 자기 자신의 주소를 반환
  - Get Blackboard : 블랙보드의 값을 받아 온다
  - Make Literal Name : 블랙보드 안의 키를 참조 // PatrolPoint(타겟)으로 바꾸어준다

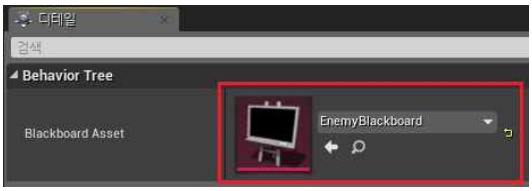


* 다음 목표 지점 설정

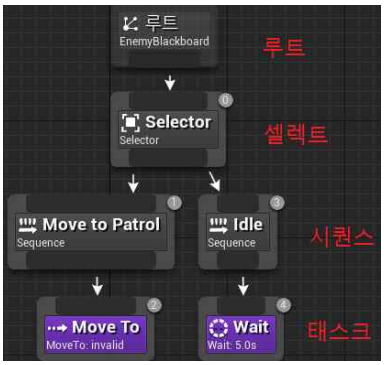


○ Behavior Tree(행동 트리)

* AI의 의사결정에 필요한 로직을 관리하며, 어떤 액션을 취할지 결정
  - BP - 디테일 - Blackboard Asset - 해당 블랙보드 설정

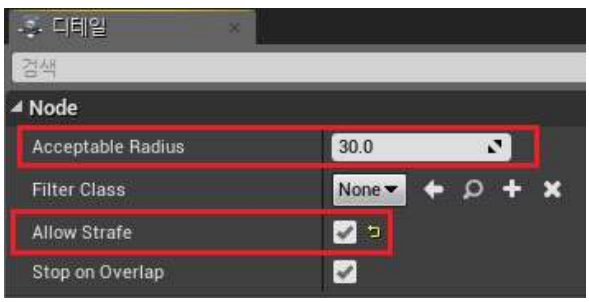


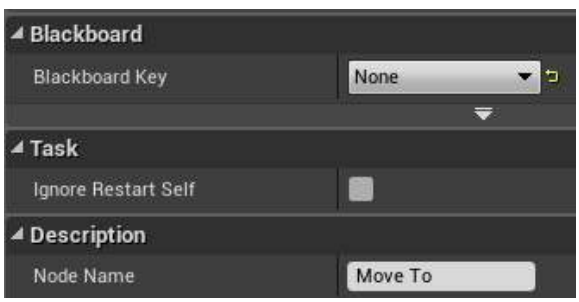
* 로직 트리 : 어떠한 액션을 취할 것인지 결정
  - 노드의 숫자 : 실행 순서



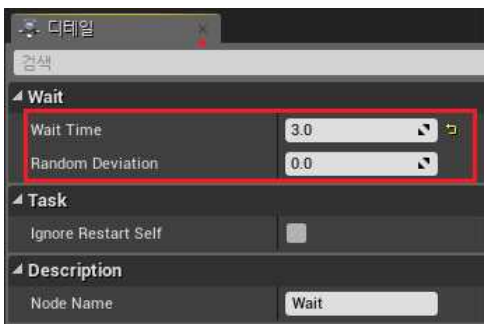
* 루트(Root) 노드 : 로직 트리의 최상단, 검은색 바를 드래그하여 새로운 노드를 추가할 수 있다
* 셀렉터(Selector) 노드 : 모든 자식 노드를 실행(왼쪽 -> 오른쪽)하며, 하나라도 실행 성공 되면 멈춤
* 시퀀스(Sequence) 노드 : 모든 자식 노드를 실행(왼쪽 -> 오른쪽)하며, 하나라도 실행 실패하면 멈춤
* 태스크(Task) 노드 : 행동 트리의 최하위의 노드 // 보라색

:: Move To : 움직임을 결정하는 태스크 노드
- Acceptable Radius : 액터가 목적지에 얼마만큼 가까워 질 때까지 움직이는지 결정
- Filter Class
- Allow Strafe : 적 캐릭터가 목적지로 움직일 때 옆으로 움직이지 않게 만듦
- Stop on Overlap





:: Wait : 멈춤 상태를 결정하는 태스크 노드
- Wait Time : 얼마나 멈출 것인지를 결정, (시간단위 : 초)
- Random Deviation : 랜덤 값 추가
  // ex) Wait Time에 3.0, Random Deviation에 1.0을 설정하면 2~4초 사이 랜덤



○ AIController

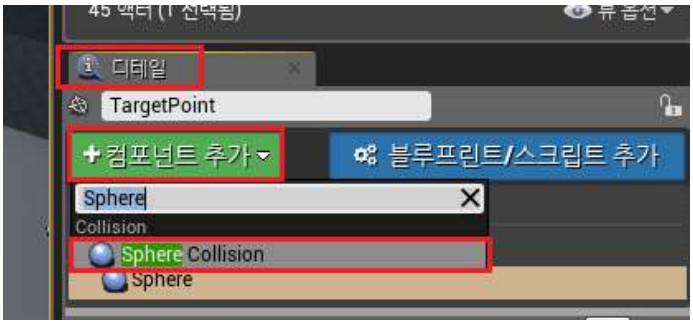
* 캐릭터와 행동 트리를 연결하는 역할, 해동 트리에서 만들어진 정보와 액션을 캐릭터에게 전달

○ Blackboard(블랙보드)

* 행동 트리에 의해 사용되는 모든 데이터를 담고있는 컨테이너

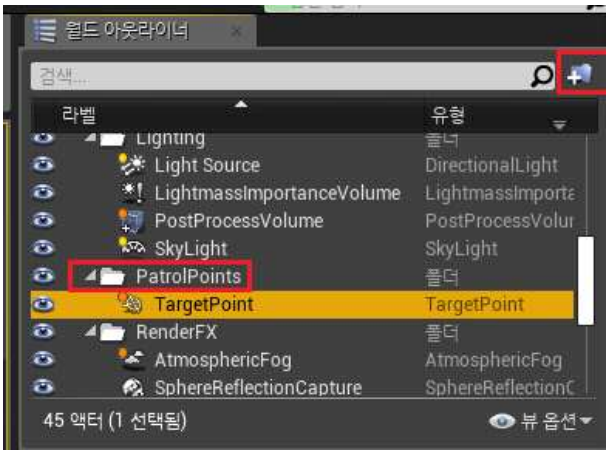
○ AI 경로 설정

* 모드 - 모든 클래스 - 타겟 포인트 생성
* 충돌 확인 구 생성 : 타겟 포인트 - 디테일 - 컴포넌트 추가 - Sphere Collision를 생성



○ 관리 폴더 추가

* 월드 아웃라이너 - 폴더 추가



○ AI 플레이어 유도탄

* 플레이어 유도