**The Laboratory Read Me**

**목차**

1. [개요](#개요)
2. [스토리 시놉시스](#스토리_시놉시스)
3. [게임/시스템 특징](#시스템특징)
4. [C++클래스](#클래스다이어그램)
5. [블루프린트](#코드리뷰)
6. [사용한 소스 출저](#사용한소스출저)

**[1.개요](#목차)**

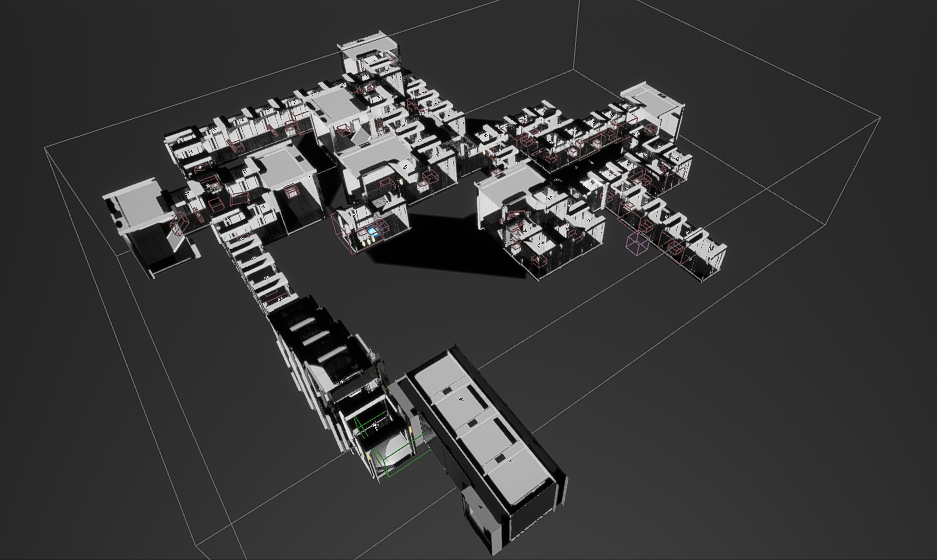
* 제작기간: 2019년 9월 ~ 2020년 6월
* 보수기간: 2020년 7월~
* 개발환경: Unreal Engine 4
* 장르: 3인칭 탈출게임
* 플랫폼: PC
* 기획의도: 복잡하게 생각하지 않고 부담 없이 가볍게 즐길 수 있는 게임을 만들고자 하였습니다. 중학생 때 광주 무등산에 올라갔다가 내려오는 길에 이상한 길로 가서 길을 잃을 뻔했던 기억이 떠올라서 미로를 탈출하는 형태의 게임으로 만들면 재밌을 것 같아서 탈출하는 게임을 만들었습니다.
* 예상되는 유저층: 모든 연령이 즐길 수 있는 난이도로 만들었습니다.
* 문서를 보실 때 하이퍼텍스트를 이용하시면 보다 편하게 보실 수 있습니다.

[**2.** **스토리 시놉시스**](#목차)

외계인들에게 납치당해 실험실로 끌려간 주인공은 같이 끌려간 박사의 도움을 받아 외계인들이 만든 로봇 중 하나를 조종할 수 있게 됐습니다. 주인공은 로봇을 조종해서 외계인들로부터 탈출하려고 합니다.

[3.](#목차) **[게임/시스템 특징](#목차)**

* 게임진행
  + 제한 시간 0이 되면 플레이어는 사망하게 됩니다.
  + 사망 시 레벨이 재시작 됩니다.
  + 길을 찾기가 힘들어서 길안내 시스템을 추가했습니다.
  + 게임내 출력되는 텍스트들은 DataTable을 이용해서 관리하고 있습니다.
  + 튜토리얼 시스템을 Gif를 이용해서 플레이어에게 시각적으로 도움을 주고자 만들었습니다.
  + 튜토리얼 시스템의 Gif 이미지는 비동기 로드 처리하였습니다.
  + 게임의 세이브/로드는 USaveGame을 이용하여 구현하였습니다.
* 플레이어
  + Item들의 정보를 저장할 인벤토리 컴포넌트를 제작했습니다.
  + Item의 종류가 적어서 팝업 UI로 나타내는 것보다 HUD에 상태를 출력시키는 것이 더 낫다고 판단되어 HUD에 아이템 정보를 출력했습니다.
  + 공격은 AnimNotify로 이벤트 분기를 만들어 일반공격과 차지공격을 구현했습니다.
  + 일반공격의 범위는 검 모델링에 BoxComponent를 이용해서 Overlap이 발생할 경우 데미지 이벤트가 발생하게 하였습니다. 검에 정확히 맞춰야 되기 때문에 공격의 조작감을 어렵게 만들었습니다.
  + 일반공격시 검에 닿을 경우 폭발할 때 일정 반경에 추가로 폭발 데미지가 발생하게 만들었습니다. 검에 닿는 물체가 많아질수록 강한 데미지가 일어나게 의도하였습니다.
  + 차지공격시 입력을 제한하고 공격 애니메이션이 끝날 때까지 움직이지 못하게 하였습니다. 그 이유는 강한 데미지를 주위에 주는 것에 따른 페널티로 구현하였습니다.
  + 미니맵은 SceneCaptureComponent2D를 이용하여 구현하였습니다. 플레이어의 머리 위에서 카메라로 촬영한 텍스쳐를 UI에 나타나게 했습니다. 플레이어에게 어느 방향으로 움직이고 있는지 힌트를 주고 미로를 탈출하게 만드는 의도로 만들었습니다.
  + 피격 효과를 Post Process를 이용하여 구현하였습니다. 통신이 끊긴다는 점을 강조하고 싶었습니다. 그래서 화면이 점점 흑백으로 바뀌고 잔상이 생기면서 주변이 잘 안보이고 밝기도 어둡게 변하게 만들어내고 싶었습니다.
* 적 캐릭터 관련
  + 로봇(Grunt)
    - 탐색, 이동, 공격 3가지 행동을 하게 만들었습니다.
    - 탐색은 주변을 스캔하여 탐지하게 만들었습니다.
    - DataTable에 5가지 스텟 데이터를 작성하고 Begin Play에서 랜덤하게 타입을 정하여 그 타입에 맞는 스텟으로 게임을 시작하게 만들었습니다.
    - 근접공격만 하지만 공격 속도가 빠른 편으로 만들었습니다.
  + 안드로이드(Gunner)
    - Ai Perception 중 sight를 이용하여 탐색을 하게 만들었습니다.
    - 플레이어를 보게 되면 추격하고 이동하며 사격합니다.
    - 사격은 총구에서 정면으로 날아가는 투사체 방식을 이용하고 안드로이드는 투사체 컨테이너 컴포넌트를 가지고 있습니다.
    - 플레이어의 어려운 공격 조작 때문에 가까이 붙을 경우 연사력을 낮췄습니다.
  + 마법사(Boss)
    - 스테이지에서 대기하기 때문에 이동을 하는 로직은 없습니다.
    - MP를 기반으로 행동하는 AI로 만들었습니다.
    - MP를 충전할 경우 상당한 시간 무방비 상태가 되는데 이 때 집중적으로 공격을 하면 깰 수 있을 패턴으로 만들었습니다.
    - 총 4가지의 공격 패턴을 만들었습니다.
      1. 순간 이동하며 번개로 공격하기
      2. 추격하며 불꽃을 날리기
      3. 순간 이동하며 불꽃 날리기
      4. 순간 이동하며 번개와 불꽃으로 공격하기
    - 순간이동은 플레이어의 위치를 기준으로 원 모양 지점에서 임의의 지점을 지정해서 그 지점에서 밑으로 ray를 이용해 바닥이 있다면 그 지점으로 이동하고 바닥이 없는 지점일 경우 다시 임의의 지점을 고르게 됩니다.
    - MP가 소모되는 기준은 공격 모션 한번 실행할 때마다 소모되며 소모량은 각각 다릅니다. 만약 MP가 부족하다면 즉시 MP를 충전하게 만들었습니다.
* 레벨 디자인



* + Stage 1 디자인
    - Stage1은 튜토리얼을 보여주는 목적이 있기 때문에 갈림길 위주로 배치해서 플레이어에게 길을 선택하게 만들어 미로에 빠져들게 만들 의도로 만들었습니다. 스테이지를 만든 저도 길을 잃을 정도로 만들었습니다.

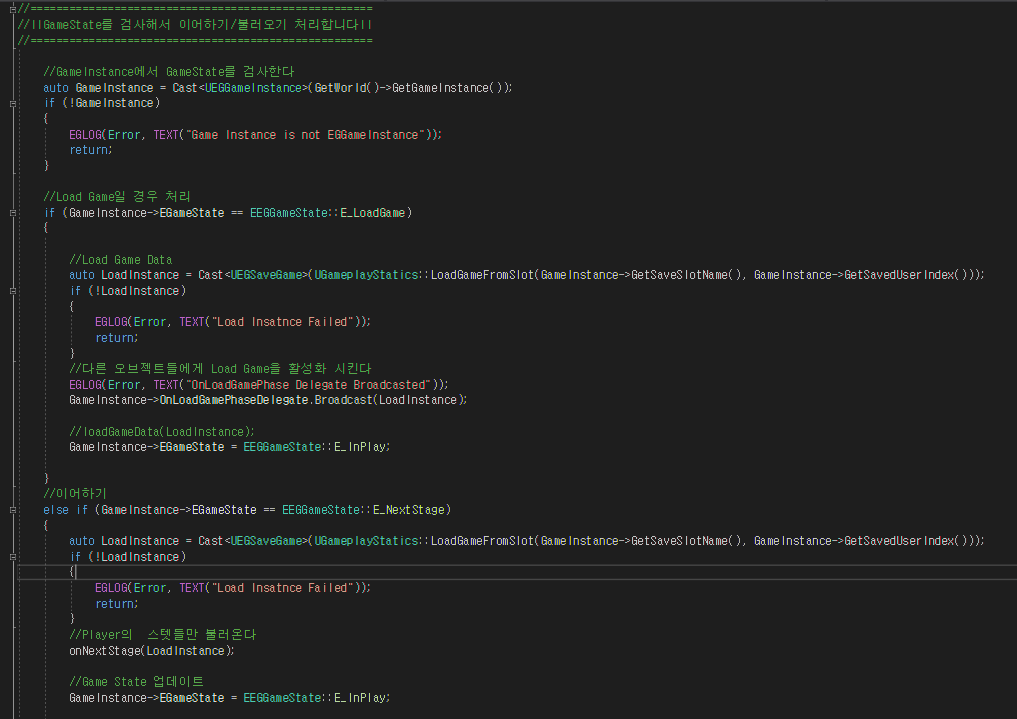
하늘, 녹색이(가) 표시된 사진

자동 생성된 설명

* + Stage 2 디자인
    - Stage1과 다르게 상대적으로 직선 구간을 줄이고 높이 변화에 중심을 뒀습니다. 적을 Stage1보다 더 많이 배치하였는데 이는 플레이어가 탈출한 것이 걸려서 외계인들이 막으러 나온 것처럼 보이게 만들고 싶었습니다.
    - 라이팅 비용이 많이 나와서 일부 구간의 라이팅을 static으로 변경해야 됐습니다.
    - Stage1에서 받은 느낌과 다르게 만들기 위해 살짝 어둡게 만들어 이펙트가 눈에 더 들어오게 만들었습니다.
    - 마지막 방으로 가는 길을 어둡고 길게 만들었는데 플레이어에게 이 앞이 마지막이라는 느낌을 받게 만들었습니다.
    - 마법사가 있는 곳을 좁게 만들었습니다. 넓게 만들어서 테스트를 해봤더니 난이도가 너무 높아져서 그랬습니다.
      1. [C++](#목차) 클래스
* 목록
  1. Player
* Player관련 클래스
  + 클래스 다이어그램

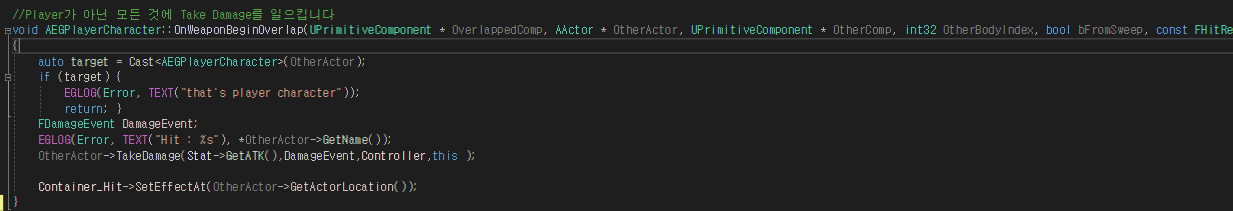


* + 클래스
    - AEGPlayerCharacter : 이 게임의 Player Character 클래스입니다.
    - UAnim\_Player : Player의 애니메이션을 담당하는 클래스입니다.
    - AEGPlayerController : Player가 사용하는 Controller 클래스입니다.
    - UMiniMapRenderComponent : 미니맵을 그려주기 위한 컴포넌트입니다.
    - UComponent\_SelfDamage : 일정 시간이 지나면 Owner Actor에게 데미지를 주는 컴포넌트입니다. 시간이 지나면서 베터리가 소모되는 것을 표현하기 위해 Player에게 부착했습니다.
    - UComponent\_Inventory : Owner Actor가 현재 가지고 있는 아이템의 정보를 저장하는 컴포넌트입니다.
    - UStatComponent\_Player: Player에게 필요한 Hp, Speed 같은 Stat 정보들을 모아둔 컴포넌트입니다.
    - USkillContainer\_PlayerHitEffect: Player의 공격에 맞은 지점에 폭발하는 이펙트를 주기 위해서 폭발 이펙트들을 저장시킨 컴포넌트입니다.
  + 코드
    - AEGPlayerCharacter
      * BeginPlay()



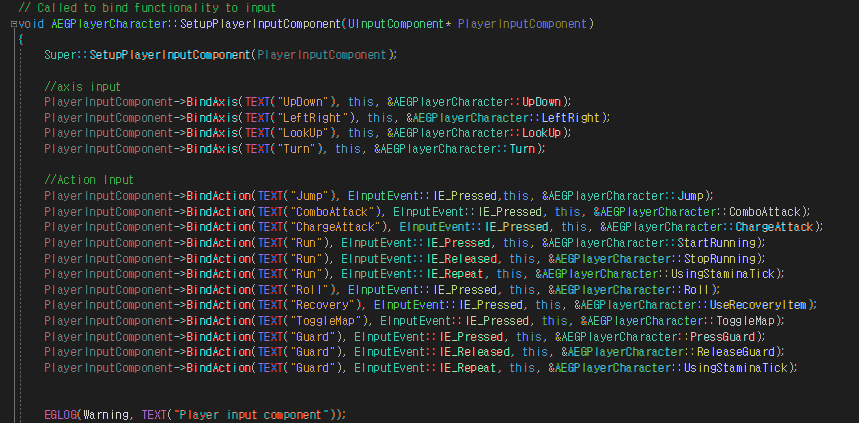
Begin Playe에서 EGGameInstance에서 GameState를 읽어들여   
LoadGame 상태라면 저장된 데이터를 불러들입니다.   
NextStage라면 Player의 정보들만 불러들여 Stage를 준비합니다.

* + - * OnWeaponBeginOverlap



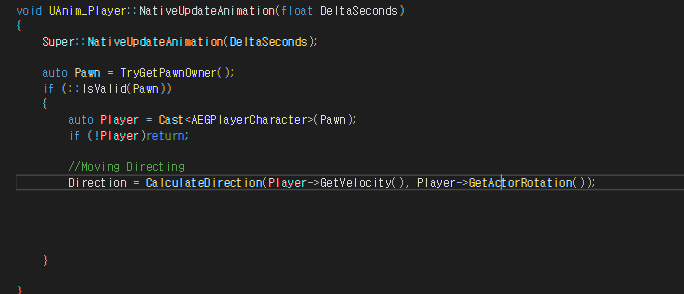
공격을 시작하는 입력이 들어오면 그 때 무기에 씌운 Collision의 profile name을 변경시켜 작동시킵니다.   
만약 플레이어가 맞았다면 넘어가고 다른 Actor들에 대해선 Take Damage를 호출하고 Container\_Hit에 저장 중인 이펙트를 칼이 맞은 위치에서 켜주면서 폭발 데미지를 입힙니다.

* + - * SetupPlayerInputComponent



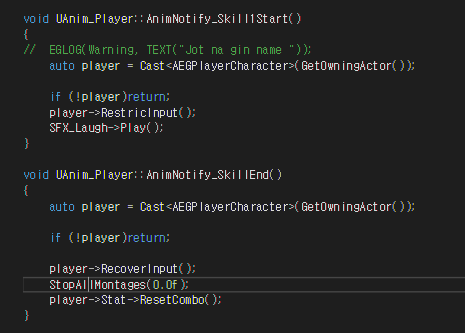
엔진의 입력에 바인딩된 함수들입니다.

* + - UAnim\_Player
      * NativeUpdateAnimation



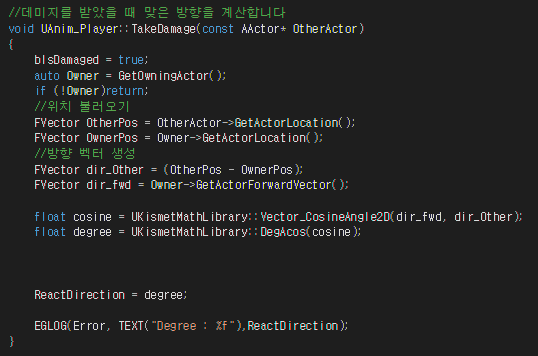
플레이어의 횡 이동 애니메이션을 표현하기 위해서 움직이는 방향을 업데이트 해줍니다

* + - * AnimNotify\_SkillStart/End



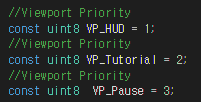
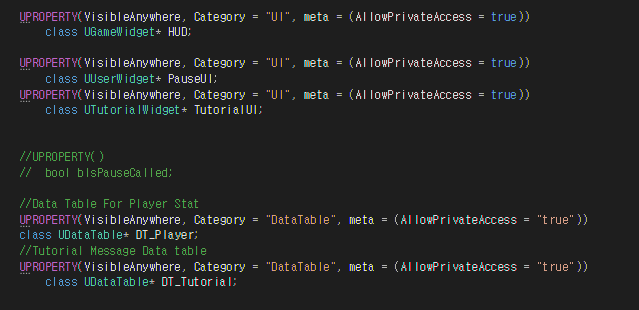
차지공격 몽타주가 실행할 때 호출이 됩니다.   
플레이어의 입력을 제한시켜 움직이지 못 하게하고 끝나면 입력을 회복시켜줍니다.

* + - * Take Damage



Player가 데미지를 입을 경우 호출됩니다. 데미지를 준 Actor와 각도를 계산해서 각도에 맞는 피격액션이 나오게끔 하였습니다.

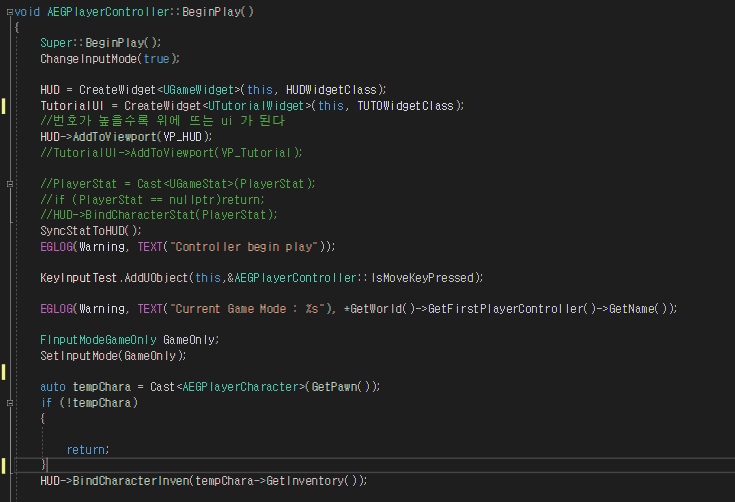
* + - AEGPlayerController
      * Header



Player Controller는 3개의 UI와 2개의 DataTable을 가지고 있습니다.

각 UI는 Viewport에 추가될 때 priority에 따라 추가됩니다.

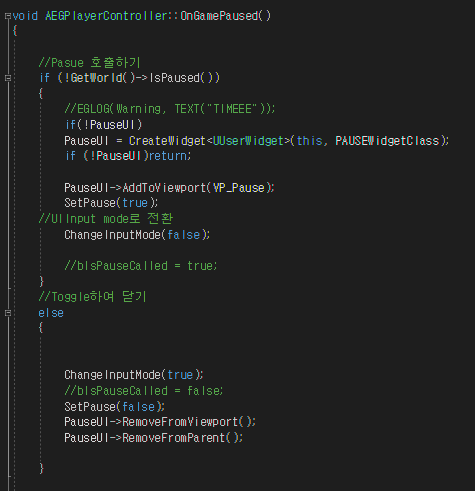
* + - * BeginPlay



HUD와 TutorialUI를 만들어주고 HUD만 Viewport에 추가해줍니다.

HUD UI와 player의 stat component와 inventory를 연동시켜줍니다.

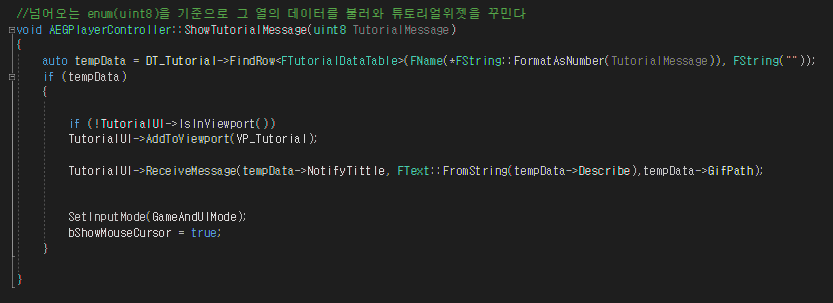
* + - * OnGamePaused



Pause UI가 생성해서 화면에 띄워주고 게임을 일시정지 시킵니다.

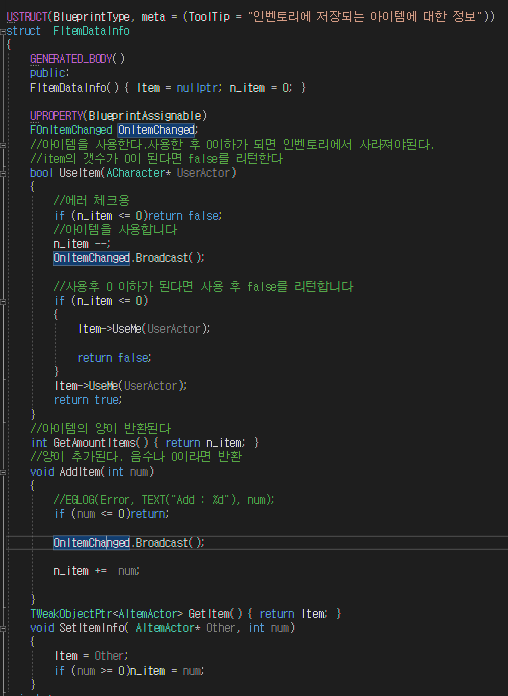
게임 입력 방식도 바꿔서 마우스로 조작이 가능하게 했습니다.

* + - * ShowTutorialMessage



ATutorialNotify와 Player가 오버랩 될 경우 ATutorialNotify에서 보낸 UENUM 정보에 따라 DataTable에서 정보를 가져와 UI에 띄워줍니다

* + - UComponent\_Inventory / FItemDataInfo
      * FItemDataInfo



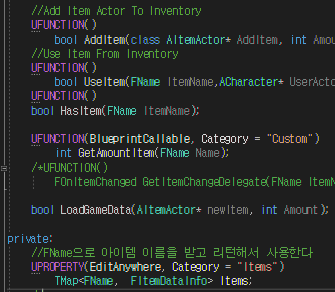
TWeakObjectPtr로 가리키는 AItemActor에 대한 정보를 나타낸 구조체입니다.

UseItem, GetAmountItmes, AddItem, GetItem, SetItemInfo 함수가 있습니다.

SetItemInfo는 이 구조체가 저장할 Item에 대한 정보를 받아올 때 사용됩니다.

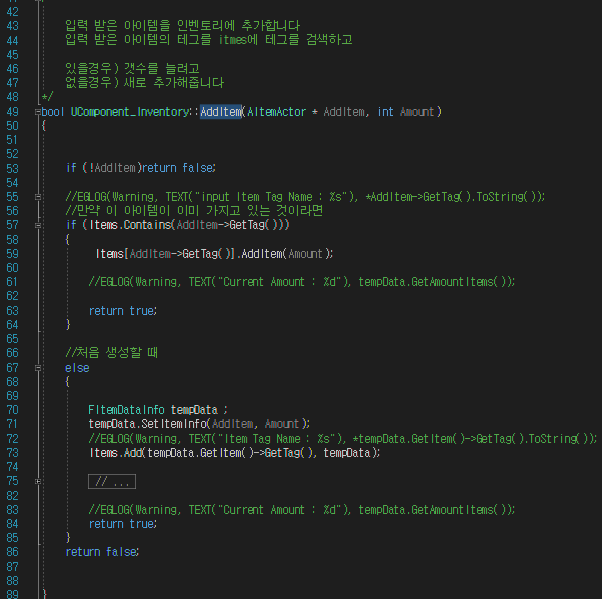
아이템의 수량이 변하게 되면 Delegate가 실행되게 만들어서 HUD에 표시되는 정보를 바꿔줍니다.

* + - UComponent\_Inventory



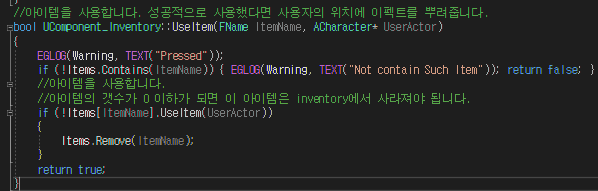
아이템을 TMap<이름, 데이터>의 형태로 저장하고 있습니다.

* + - * AddItem



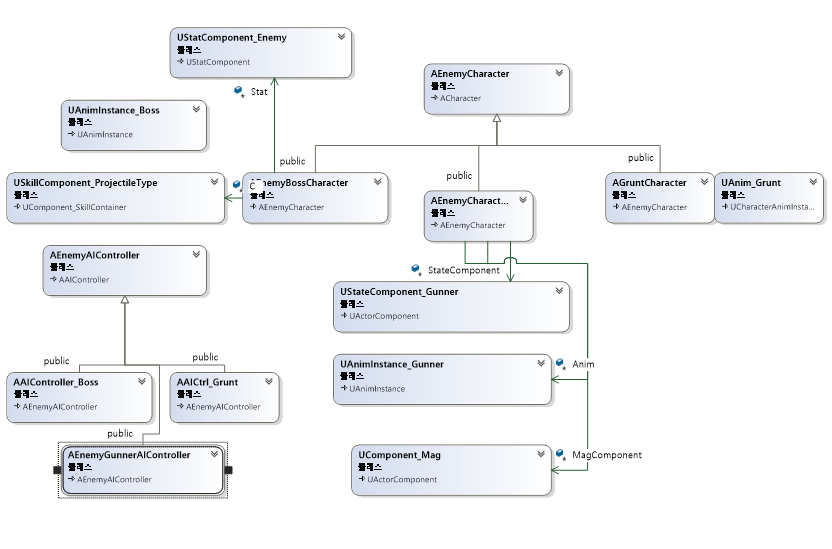
Item에 등록시킨 Tag를 통해서 Item의 이름을 정했고 Item을 가지고 있는지 검사하고 없다면 새로 추가하고 있다면 개수를 늘려줍니다.

* + - * UseItem



사용 이펙트가 있는 Item이라면 이펙트를 재생하고 ItemData에서 개수를 줄여주는 작업을 합니다.

* Enemy 계열 클래스
  + 다이어그램



* + 클래스
    - AEnemyCharacter : Enemy Character들의 기본 클래스입니다.
  + sf

[5. 블루프린트](#개요)

* 목록

[6.사용한 소스 출저](#목차)