

언리얼 프로그래밍 Part2-2

제목: 캐릭터와 입력 시스템

**강의 내용 : C++ 액터와 마네킹 캐릭터의 제작

강의 내용

C++ 액터와 마네킹 캐릭터의 제작



첫 번째 강의에서 등장한 마네킹 캐릭터를

**강의에서 다루는 게임 프레임워크 요소

강의에서 다루는 게임프레임워크 요소

게임	월드	모드	상태	
기믹	트리거	스폰	물리	
플레이어	입력	카메라	HUD	상태
폰	이동	모션	액션	위젯
데이터	애셋	테이블	설정	저장
인공지능	길찾기	BT		

-전체 프레임워크에서의 입력과 폰에 대한 기초적인 내용을 다룰거임.

**강의 목표

강의 목표

- 액터와 컴포넌트 개념의 이해
- 블루프린트로 확장 가능한 프로퍼티 설계
- 언리얼 엔진의 폰과 캐릭터 시스템의 이해
- 언리얼 엔진 5이 제공하는 향상된 입력 시스템의 활용

이득우의 언리얼 프로그래밍 Part2 - 언리얼 게임 프레임워크의 이해
이득우

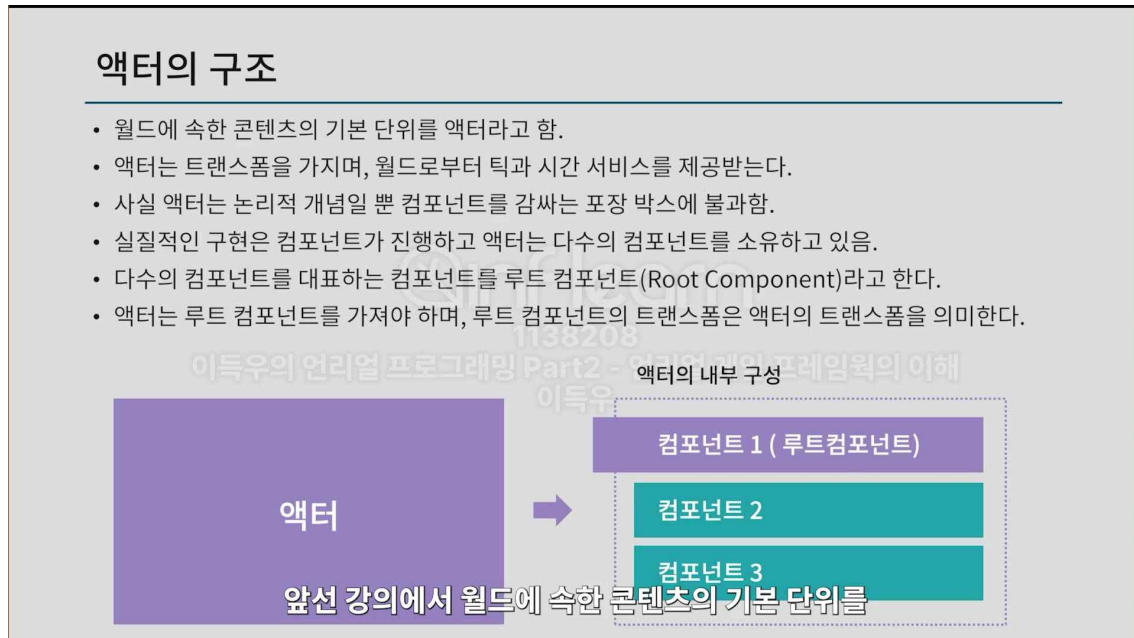
이를 바탕으로 C++ 액터를 직접 생성해보도록 하겠습니다

-C++액터에서 블루프린트로 확장해볼것임 그리고 액터로 확장하여 플레이어가 빙의해 조종하는 폰과 캐릭터 시스템을 다룰것임.

-향상된 입력시스템을 이용하여 우리가 제작한 캐릭터를 조종해볼거임.

***액터와 컴포넌트

**액터의 구조



**실습

-DefaultSceneRoot는 임시적으로 생성한 아무 기능이 없는 컴포넌트다.

**C++액터에서 컴포넌트의 생성

C++ 액터에서 컴포넌트의 생성

- 컴포넌트는 언리얼 오브젝트이므로 UPROPERTY를 설정하고 TObjectPtr로 포인터를 선언한다.
 - 언리얼 5버전부터 헤더에 언리얼 오브젝트를 선언할 때 일반 포인터에서 TObjectPtr로 변경
- 컴포넌트의 등록
 - CDO에서 생성한 컴포넌트는 자동으로 월드에 등록된다.
 - NewObject로 생성한 컴포넌트는 반드시 등록절차를 거쳐야 한다. (예) RegisterComponent)
 - 등록된 컴포넌트는 월드의 기능을 사용할 수 있으며, 물리와 렌더링 처리에 합류한다.
- 컴포넌트의 확장 설계
 - 에디터 편집 및 블루프린트로의 승계를 위한 설정
 - UPROPERTY에 지정자(Specifier)를 설정할 수 있다.
- 컴포넌트 지정자
 - Visible / Edit : 크게 객체타입과 값타입으로 사용
 - Anywhere / DefaultsOnly / InstanceOnly : 에디터에서 편집 가능 영역
 - BlueprintReadOnly / BlueprintReadWrite : 블루프린트로 확장시 읽기 혹은 읽기쓰기 권한을 부여
 - Category : 에디터 편집 영역(Detail)에서의 카테고리 지정

먼저 액터를 선언하기 전에 알아야 될

-CDO : 클래스 디폴트 오브젝트

-컴포넌트 지정자로 객체타입인 경우 VIsible/Anywhere을 자주 사용한다

-컴포넌트 지정자로 값타입인 경우 Edit/Anywhere을 자주 사용한다

-컴포넌트 같은 경우 객체타입으로 VIsible/Anywhere을 사용하면 된다.

-두개의 StaticMesh Component를 Fountain 액터에다가 추가해보자(실습)

-만들어진 c++클래스를 상속해서 별도의 블루프린트로 액터를 만들 수 있다.

-즉 c++로 기반을 충분히 다져놓고 이후에 추가적인 로직은 블루프린트를 사용해서 필요한 만큼만 최소 한도로 확장하도록 설계하는 것이 앞으로 게임 제작을 효과적으로 진행하는데 있어서 유용하게 사용될 수 있다.

**실습

-만들어진 블루프린트의 경우에 C++로 기반을 충분히 다져놓고 추가적인 로직은 블루프린트를 사용하여 필요한 만큼만 최소 한도로 확장하도록 설계하는 것이 게임 제작에 효과적으로 진행하는데에 있어서 유용하게 사용될 수가 있다.

***캐릭터의 제작

**폰의 기능과 설계

폰의 기능과 설계

- 폰은 액터를 상속받은 특별한 액터이며, 플레이어가 빙의해 입출력을 처리하도록 설계되어 있음.
- 폰은 길찾기를 사용할 수 있으며, 일반적으로 세 가지 주요 컴포넌트로 구성된다.
 - 기믹과 상호작용을 담당하는 충돌 컴포넌트 (루트컴포넌트)
 - 시각적인 비주얼을 담당하는 메시 컴포넌트
 - 움직임을 담당하는 컴포넌트
- 컴포넌트 중에서 트랜스폼이 없이 기능만 제공하는 컴포넌트를 액터컴포넌트라고 한다.

폰의 내부 구성

폰

충돌 컴포넌트 (루트컴포넌트)

비주얼 컴포넌트

움직임 컴포넌트

액터를 상속받은 특별한 액터를 의미합니다

-충돌 컴포넌트, 메시 컴포넌트, 무브먼트 컴포넌트 세가지 요소를 기본적으로 사용해서 폰을 제작하는 것이 정석

-움직임의 경우 트랜스폼 없이 기능만 제공하는 것을 액터 컴포넌트, 트랜스폼이 있는 것은 씬 컴포넌트라고 한다.

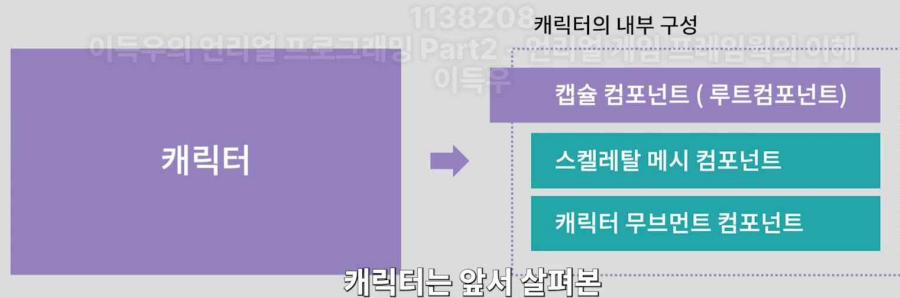
-루트 컴포넌트는 언제나 기믹과 상호작용을 담당하는 충돌 컴포넌트로 지정이 된다.

-우리가 폰을 직접 만들진 않고 캐릭터 구조로 넘어갈거임.

**캐릭터의 기본 구조

캐릭터의 기본 구조

- 캐릭터는 인간형 폰을 구성하도록 언리얼이 제공하는 전문 폰 클래스를 의미한다.
- 캐릭터는 세 가지 주요 컴포넌트로 구성되어 있다.
 - 기믹과 상호작용을 담당하는 캡슐 컴포넌트 (루트컴포넌트)
 - 애니메이션 캐릭터를 표현하는 스켈레탈 메시 컴포넌트
 - 캐릭터의 움직임을 담당하는 캐릭터 무브먼트(CharacterMovement) 컴포넌트



-실습(마네킹 캐릭터 제작)

(1)플레이어에 관련된 기본 기능과 불필요한 함수 모두 제거-->AABCharacterBase() 밑에 다 지우면 됨.

(2)Character.h를 보면 252번째줄을 보면 컴포넌트들이 이미 구축되어 있지만 private이기에 직접 접근이 불가능하고 상속 받은 클래스는 GetMesh()와 같은 함수를 통해 접근이 가능함. 또한 메시와 관련된 다양한 설정값들을 생성자 코드에서 설정하면 된다.

(3)ABCharacterbase를 만들고 ABGameMode의 DefaultPawn을 수정후 컴파일하여 실행하면 화면이 멈춰있을텐데 이는 카메라를 아직 설정하지 않아서 그렇다.

(4)카메라 기능의 경우 NPC는 필요없고 카메라만 필요하기 때문에 ABCharacterPlayer클래스에 카메라 컴포넌트와 기능을 추가하자.

(5) UPROPERTY에서 설정 가능한 지정자중 Meta = (AllowPrivateAccess = "true")는 우리가 Private로 선언된 어떤 언리얼 오브젝트의 객체들을 블루프린트에서도 접근 가능하도록 해주는 지시자이다.

***입력 시스템 개요

**입력시스템의 동작 방식

입력시스템의 동작 방식

- 플레이어의 입력은 컨트롤러를 통해 폰으로 전달됨.
- 입력을 컨트롤러가 처리할 수도, 폰이 처리할 수도 있는데, 일반적으로는 폰이 처리하도록 설정
 - 예) GTA같은 다양한 사물에 빙의하는 게임의 경우 폰이 유리

먼저 언리얼 엔진의 입력시스템의 동작 방식을 제가 정리해 봤는데요

-플레이어가 주는 입력은 위 사진처럼 플레이어 컨트롤러를 통해 Pawn으로 전달된다.(즉 우선순위는 플레이어 컨트롤러가 가진다.)

-따라서 입력 처리를 플레이어 컨트롤러가 처리할수도 있고, Pawn이 처리할 수도 있다.(우리가 어떤 게임을 만드느냐에 따라서 지정을 해 줄 수 가 있다.

-GTA같은 다양한 사물에 빙의하여 조종하는 게임을 만든다고 가정할 때 입력에 대한 모든 처리를 플레이어 컨트롤러가 담당한다면 코드가 굉장히 방대해진다. 이러면 구조상 좋지 않기 때문에 각 빙의를 담당하는 자동차라던지 캐릭터에 대해서는 각각의 입력 처리를 빙의를 당하는 Pawn에서 구현해주는 것이 코드가 분산되어 보다 효과적으로 코드 구현이 가능하다.

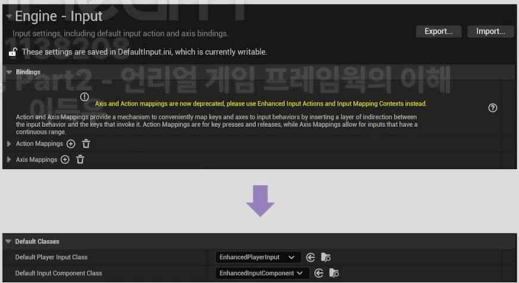
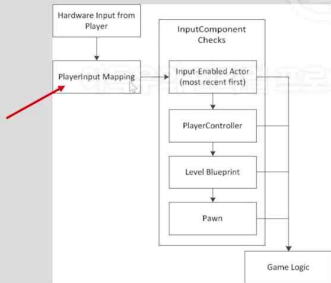
-입력 방식에 대한 자세한 방법은 위 사진의 사이트를 참고하여 한번 읽어보면 좋다.

***향상된 입력 시스템(5.1버전에서 추가 됨)

**향상된 입력시스템(Enhanced Input System)

향상된 입력시스템(Enhanced Input System)

- 기존 입력시스템을 대체하고 언리얼 5.1부터 새롭게 도입
- 사용자의 입력 설정 변경에 유연하게 대처할 수 있도록 구조를 재수립
- 사용자 입력 처리를 네 단계로 세분화하고 각 설정을 독립적인 애셋으로 대체



플레이어의 입력(Input)을 설정하는 과정에서

-앞서 설명한 입력시스템에는 한가지 단점이 존재하는데 플레이어의 입력(Input)을 설정하는 과정에서 처리하기 전에 Input을 설정하다 보니 고정되어 있어 나중에 사용자의 입력설정 변경에 유연하게 대처할 수가 없었다.

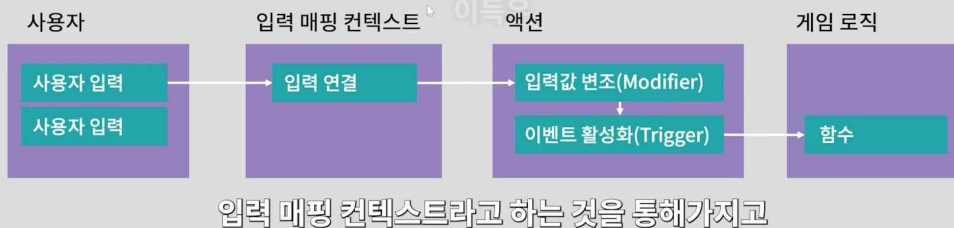
-위에 말한 문제를 해결하기 위해서 향상된 입력 시스템이라는 것을 도입하였는데 플레이어의 최종 입력을 게임 로직에서 진행하도록 시스템의 구조를 조금 변경하였다. 또한 사용자의 입력처리를 4 단계로 세분화하고 각 설정을 독립적인 애셋으로 대체하여 다양한 상황에 대해서 유연하게 대처할 수 있도록 효과적으로 구조를 만들었다.

-실제로 엔진의 프로젝트 설정에 입력으로 위 사진에서 위에 사진은 이전 설정이고, 밑에는 최근 설정인데 밑에 사진에서는 Default Class로 Enhanced라고 되어있는 새로운 클래스들을 사용하도록 지정이 되어 있다.

**향상된 입력시스템의 동작 구성

향상된 입력시스템 동작 구성

- 사용자의 입력 데이터를 최종 함수에 매핑하는 과정을 체계적으로 구성
- 플랫폼에 따른 다양한 입력 장치의 설정
 - 예) 게임패드용 입력 매핑 컨텍스트, 키보드용 입력 매핑 컨텍스트
- 입력 값의 변경
 - 예) AD/WS 입력값을 Y축과 X축으로 변경, 값 반전의 처리
- 이벤트 발생 조건의 상세 설정
 - 예) 일반 버튼인가? 축 이동인가? 일정 이상 눌러야 하는가?



-동작방식

- (1) 사용자가 입력 진행
- (2) 입력 매핑 컨텍스트를 통해 어떤 입력과 연결됨
- (3) 이 입력을 연결했을 때 수행하는 주체가 액션임
- (4) 우리가 지정한 액션에 이 사용자 입력이 연결이 됨.
- (5) 이 액션에 대해서는 추가적인 설정을 통해 어떻게 들어온 입력값을 재가공하는지 그러한 입력값을 통해서 어떻게 이벤트를 활성화 할지 설정을 진행 함.
- (6) 설정을 진행 한 후에 우리가 만든 게임 로직에서 지정한 특정 함수와 매핑을 진행함
- (7) 이러한 전체적인 4단계를 거쳐 보다 유연하게 입력 설정이 가능하다.

-두번째 입력 매핑 컨텍스트에서는 게임패드용 매핑 컨텍스트라던지 키보드용 입력 매핑 컨텍스트들을 런타임에서 자유롭게 바꿀 수 있도록 설계가 가능하다.

-액션에 대해서는 우리가 이동할 때 많이 사용하는 A,D,W,S 입력값을 보통은 Y축과 X축으로 변경을 하거나 반전으로 처리할 때가 있는데 이 변조를 통해 우리가 조종할 수가 있다.

- 일반 버튼인지 축이동인지 아니면 일정 이상 눌러야 되는 어떤 다양한 액션처리들에 대해서는 이벤트 활성화 트리거 액션을 통해 조종해서 이벤트를 발생시킬 수가 있다.(과거에는 이러한 것들을 다 게임 로직에서 처리함) 이제는 별도의 액션을 통해 가지고 관리할 수 있게 함으로써 게임 로직에 부담을 덜어주고 런타임에서 우리가 원하는 설정들을 자유롭게 변경할 수 있도록 입력 시스템을 향상 시켰다.

**캐릭터 입력 실습

-입력의 경우 UPROPERTY 지정자로 EditAnywhere을 설정하는데 이 에셋들의 경우 다른 에셋으로 변경할 수 있도록 설계하기 위해 EditAnywhere을 로 설정했다.

-한가지 매핑 컨텍스트와 3가지 액션들이 있는데 이것은 3인칭 템플릿에서 제공하는 것을 그대로 사용한다.

-두가지 함수를 추가함(각 입력 액션에 대해서 매핑된 함수들을 직접 구현)

-Move modifier 수정



**정리

언리얼 캐릭터와 향상된 입력 시스템의 이해

1. 액터와 컴포넌트 개념의 이해
2. 컴포넌트 속성에 지정자를 추가해 블루프린트로 확장하는 방법의 이해
3. 폰과 캐릭터를 구성하는 3대 구성 요소
4. 언리얼 5.1에서 추가된 향상된 입력 시스템의 사용 방법

inflearn

1138208

이득우의 언리얼 프로그래밍 Part2 - 언리얼 게임 프레임워크의 이해
이득우

이번 강의에서는 언리얼 캐릭터를 제작해보고