〈Cinemachine〉

{개요}

– 목적: Cinemachine을 이용해서 카메라 동작을 보다 유연하고 자연스럽게 구현하는 기본 개념과 실습
– 대상: Unity 사용 경험은 있으나, 카메라 제어 / Cinemachine은 처음이거나 기본적인 수준인 개발자
– 준비물: Unity 최신 버전, Cinemachine 패키지 설치된 프로젝트, 간단한 씬 (플레이어/오브젝트/배경)

1. Cinemachine 이란 무엇인가

– 가상 카메라(Virtual Cameras)를 사용하여 여러 카메라 설정을 유연하게 전환 가능

– Unity 기본 카메라(Camera) 의 기능을 확장

– 카메라 움직임, 흔들림(shake), 트랙 따르기(follow), 특정 상태에서의 카메라 전환 등

게임용

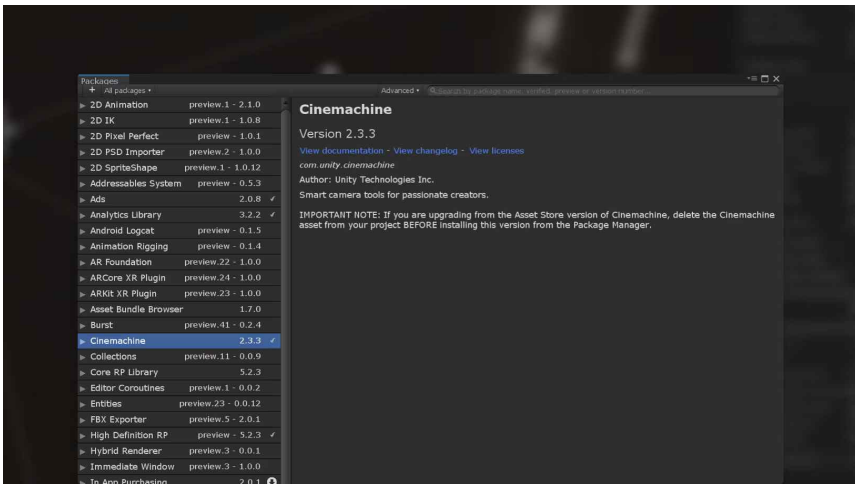
시네머신은 게임 개발 속도를 높입니다. 팀원들은 값비싼 카메라 로직 개발에서 벗어나 플레이 모드에서 설정을 저장하면서 새로운 아이디어를 즉석에서 반복하고 프로토타입을 제작할 수 있습니다. 1인칭 슈팅 게임에서 팔로우 캠, 2D에서 3D, 실시간 플레이에서 컷씬에 이르기까지 시네머신 카메라 모듈은 수년에 걸쳐 완성도를 높여왔습니다.

2. Cinemachine 패키지 설치

– Unity 에디터〉 Window〉 Package Manager〉 ‘Cinemachine’ 검색 〉 설치

– 혹은 Asset Store에서 설치 가능

– 설치 후, 상단 툴바나 메뉴에서 Cinemachine 관련 툴이 보이는지 확인



3. 주요 구성 요소

요소	역할/특징
Cinemachine Brain	실제 Unity 카메라 오브젝트에 붙어서, 어떤 가상 카메라(Virtual Camera)가 활성(active)인지 감지하고 장면에 보여줄 카메라 출력을 결정
Virtual Camera	씬 내에서 카메라의 위치, 회전, follow/LookAt 설정 가능
FreeLook Camera	타겟 중심으로 원형 또는 수평/수직 축을 따라 회전 가능한 카메라
Blend / Transitions	여러 Virtual Camera 간의 부드러운 전환 가능
Dolly Camera / Track	카메라 트랙을 따라서 움직이게 함
ClearShot, State-Driven, etc.	게임 상태 혹은 트리거 등에 따라 자동으로 카메라 전환을 관리하거나 여러 카메라 중 가장 좋은 숏(Shot)을 선택함

4. Virtual Camera 기본 사용법

- Virtual Camera 추가

  메뉴: GameObject〉 Cinemachine〉 Virtual Camera

- Follow, LookAt 타겟 설정
- Body / Aim 종류 설정 (예: Transposer, Composer, Do Nothing 등)
- Lens 설정 (Field of View, Near / Far clipping plane 등)

5. 카메라 트래킹 & LookAt

- Follow: 오브젝트를 따라다니는 카메라
- LookAt: 지정된 오브젝트를 바라보는 카메라
- Body / Aim 컴포넌트 조합으로 부드러운 트래킹/회전 구현

[실습 예제 코드]

```
using UnityEngine;
using Cinemachine;

public class CameraController : MonoBehaviour
{
    public CinemachineVirtualCamera vCam;
    public Transform followTarget;
    public Transform lookAtTarget;

    void Start()
    {
        if (vCam == null) vCam = GetComponent<CinemachineVirtualCamera>();
        if (vCam != null)
        {
            vCam.Follow = followTarget;
            vCam.LookAt = lookAtTarget;
        }
    }
}
```

6. 블렌딩 & 트랜지션 (카메라 전환 효과)
- Blend List Camera / State-Driven Camera 사용
- 우선순위(priority) 설정: 여러 Virtual Camera가 있을 때 우선순위가 높은 쪽이 활성됨
- Blend 설정: 전환 곡선(curve), 시간(duration) 등

[실습 아이디어]
- 플레이어가 특정 지역에 들어가면 다른 Virtual Camera로 전환하면서 줌(zoom) 인 또는 회전 효과 주기

7. 특수 카메라 기능들
- Dolly / Track: 트랙을 따라 카메라가 움직이면서 씬을 따라가게 함
- ClearShot Camera: 여러 카메라 배치 → 가장 좋은 뷰 선택
- FreeLook: 자유롭게 회전 가능한 카메라 컨트롤 (예: Third-person 시점)
- Target Group Camera: 여러 타켓을 한 화면에 담기

8. 스크립트에서 Cinemachine 제어하기

- 가상 카메라의 Lens, Follow, LookAt, Body/Aim 설정 변경
- 특정 이벤트에 따라 Priority 조정하여 카메라 전환
- 트랙(trigger) 또는 애니메이션 상태에 따른 카메라 활성화

[예제 코드]

```
using UnityEngine;
using Cinemachine;

public class CameraSwitcher : MonoBehaviour
{
    public CinemachineVirtualCamera cam1;
    public CinemachineVirtualCamera cam2;

    public void SwitchToCam2()
    {
        cam1.Priority = 0;
        cam2.Priority = 10;  // 높은 우선순위로 활성화
    }

    public void ResetCam()
    {
        cam2.Priority = 0;
        cam1.Priority = 10;
    }
}
```

9. 실습 예제: “플레이어 추격 카메라 + 이벤트 전환”

[목표]
– 일반 상태에서는 플레이어를 뒤에서 따라가는 카메라
– 보스 지역 진입 시, 보스가 중앙에 보이도록 카메라 위치/회전 전환
– 전투 종료 시 기본 카메라로 복귀

[단계]
(1) 기본 Virtual Camera 설정: Follow = 플레이어, LookAt = 플레이어
(2) 보스 구역 전환용 Virtual Camera 설정: 특정 위치 / 각도를 가짐
(3) Collider 또는 트리거 설치 → OnTriggerEnter로 카메라 전환 스크립트 실행
(4) Blend 시간과 우선순위 조절

10. 팁 & 주의사항
– Virtual Camera 수 너무 많으면 우선순위 / 전환 관리 복잡해짐
– Blend 시간과 카메라 전환 효과 조정 시 플레이어에게 어색함이 없는가 테스트
– 성능: Dolly트랙, LookAt, Follow 등의 업데이트 빈도(Standby Update) 설정해서 최적화 고려
– 2D 게임에서는 직교 카메라(orthographic) 설정 유의