|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **주차** | 2주차 | **기간** | | 2024.09.07  ~ 2025.09.13 | **지도교수** | (서명) | |
| **이번주 한일 간단 요약** | 1. 언리얼 C++ 외부 강의 시청  2. 기획 회의 및 피드백 | | | | | | |
| **상세 내용** | 1.언리얼 C++ 외부 강의 시청  1-1. 언리얼 C++ 기본 개념 학습  언리얼 C++에서는 .h 파일에 클래스 선언과 매크로를, .cpp 파일에 로직을 작성하면서, 게임 시작 시 한 번 실행되는 BeginPlay()와 매 프레임 호출되는 Tick(float DeltaTime)으로 동작하며,    여기서 DeltaTime은 프레임 간 경과 시간으로, 이를 곱해 주면 컴퓨터 성능이나 FPS와 관계없이 일정한 속도로 움직임을 만들 수 있다.      또한 UPROPERTY()는 변수를 에디터나 블루프린트에 노출하는 역할을 하면서 동시에, EditAnywhere는 수정 가능, VisibleAnywhere는 확인만 가능하다는 것을 알게 되었다.    1-2. C++/언리얼 기본 문법 학습  블루프린트에서 사용하던 SetActorLocation과 GetActorLocation 같은 이동/회전 노드가 실제로는 AActor 클래스의 C++ 함수임을 학습했고,    AActor 클래스 객체의 현재 위치를 가져와 이동 거리(platformVecotr) 계산 후 앞에서 설명한 DeltaTime의 경과 시간과 계산해서 새 위치로 적용  GetActorLocation()으로 현재 위치를 가져와 FVector로 이동 벡터를 계산하고 DeltaTime을 곱해 프레임 독립적으로 이동한 뒤 SetActorLocation()으로 새 위치를 적용하는 방식을 익혔으며,    회전 속도(RotationVelocity)에 DeltaTime을 곱해 프레임 독립적인 회전량을 계산하고,  AddActorLocalRotation()으로 로컬 축 기준 액터 회전에 적용  AddActorLocalRotation()을 이용해 로컬 축 기준으로 회전 값을 추가하고 독립적인 회전도 구현할 수 있음을 배웠다.  언리얼 C++에서 BeginPlay()와 Tick(DeltaTime)으로 액터를 제어하고, 블루프린트에서 사용하던 문법을 C++에 사용해서 DeltaTime과 함께 구현하는 방법을 학습했다.  AActor 클래스를 이용한 간단한 장애물 게임 :  <https://youtu.be/YZ_14yGUNRk>  2. 기획 회의 및 피드백  토요일 오후 9시에 기획 회의에 참여하여 피드백을 진행했고, 이를 통해 향후 진행될 졸업 작품의 큰 틀을 정립했으며, 다음 주부터 프로토타입 제작을 시작할 계획이다.  텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진  AI 생성 콘텐츠는 정확하지 않을 수 있습니다. | | | | | | |
| **다음주차** | 3주차 | | **다음기간** | | | | 2024.09.14 ~ 2024.09.20 |
| **다음 주 할 일** | 1. C++ 외부 강의 시청 및 문법 학습  2. 언리얼 엔진으로 프로토타입 제작  3. 팀원 들 간의 회의 참가 및 피드백 | | | | | | |
| **지도 교수의**  **피드백** |  | | | | | | |