|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **주차** | 9주차 | **기간** | 2023.8.21~ 2023.8.27 | **지도교수** | (서명) |
| 이번주 한일 요약 | - 게임 서버 공부(JobQueue)  - 언리얼 멀티플레이 공부(유데미 강의) | | | | |

<상세 수행내용>

**JobQueue**: 특정 작업이나 태스크를 순차적으로 처리하기 위해 줄세우는 구조

**비동기 처리**: 사용자의 요청이나 데이터 처리와 같은 작업을 즉시 처리하지 않고 나중에 순차적으로 처리하기 원할 때 사용.

**로드 분산**: 시스템이 과부하 상태일 때, JobQueue를 사용하여 작업을 나중에 처리할 수 있도록 임시로 저장해두고, 시스템의 부하가 감소하면 작업을 처리하게 할 수 있음.

**분산 시스템**: 여러 서버나 노드에서 작업을 처리해야 할 때, JobQueue를 사용하여 각 서버나 노드에 작업을 분배하게 할 수 있음.

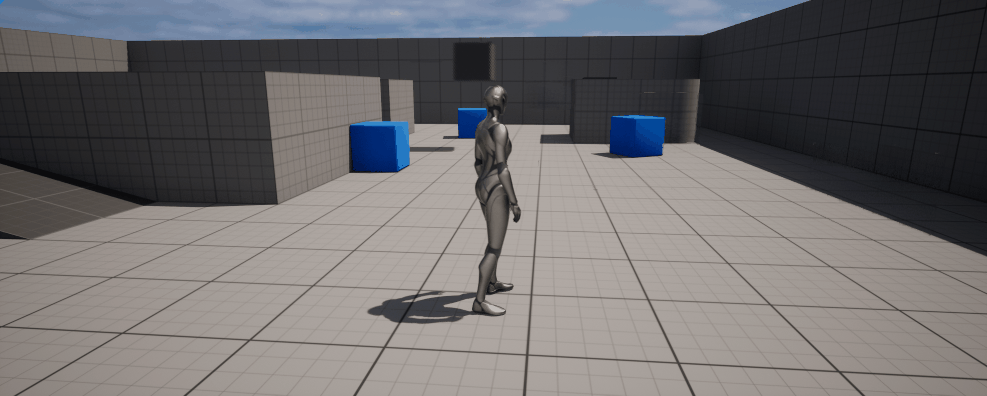
**Command 패턴**: 요청을 객체의 형태로 캡슐화

**Command 패턴의 장점**:

* 수행할 작업을 객체로 만들어서 다른 작업과 쉽게 교환할 수 있다.
* 작업의 실행을 연기하거나 큐에 넣고, 요청에 따라 로그에 기록하며, 취소 가능한 연산을 지원한다.
* 호출자와 수신자 사이의 결합도를 줄여, 기존 코드의 변경 없이 새로운 명령을 추가할 수 있다.

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

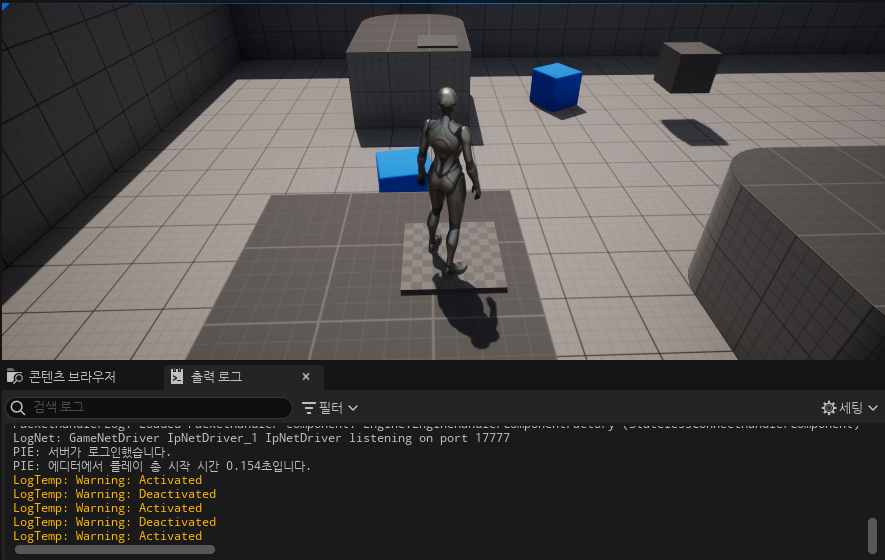
자동 생성된 설명



FVector를 활용하여 MovingPlatform이 처음 위치와 TargetLocation 사이를 왕복하도록 설계.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명



**UBoxComponent**를 사용해서 발판을 트리거로 설정해 발판에 올라갔을 때는 Activated, 내려올 때는 Deactivated가 출력되도록 함.

**OnOverlapBegin:** 이 이벤트는 두 개의 컴포넌트가 오버랩(즉, 공간적으로 겹칠 때) 시작할 때 발생

**OnOverlapEnd:** 오버랩 중이던 두 컴포넌트 중 하나가 다른 컴포넌트의 오버랩 영역에서 벗어날 때 이 이벤트가 발생

|  |  |  |  |
| --- | --- | --- | --- |
| **문제점 정리** |  | | |
| **해결방안** |  | | |
| **다음주차** | 10주차 | **다음기간** | 2023.8.28 ~ 2023.9.3 |
| **다음주 할일** | - 게임 서버 공부(데이터베이스 연동)  - 언리얼 멀티플레이 공부(유데미 강의) | | |
| **지도 교수**  **Comment** |  | | |