|  |  |  |  |
| --- | --- | --- | --- |
| [Cranding(크랜딩)] | | | |
| **주차** | **2024.01.07~ 2024.01.13** | **작성자** | **정경서** |
| **주간 회의** | | | |
| **회의 날짜** | | 2024.01.10 | |
| 1. 서버 - 클라 통신  * 시간에 관한 정보는 서버 -> 클라: 인게임에서 시간 UI로 띄워줘야함 * 서버에서 상태에 대한 정보가 넘어오면 클라에서 상황에 맞는 애니메이션 가능  1. 플레이어 애니메이션 정하기  * 이동 - Run\_01 * idle - Sneak\_Idle\_02 * 공격/기절 애니메이션 -> 믹사모 찾아보기 * 보스 처치/행성 탈출/아이템 상자 발견 ~> Pickup\_Item  1. 서버 동기화, 시간 흐름 관리 목적 씬 필요  * 평지에 플레이어 하나 띄운 씬 제작해서 main 브랜치에 올리기 -> 경서 | | | |
| **이번주 한 일** | | | |
| 1. 신동엽(서버)   클라이언트 네트워크 클래스  - 서버와 연결을 위해 클라이언트 부분에 네트워크 클래스를 따로 설계  -  - 서버의 IOCP 모델과 연결하기 위해 논블로킹 방식 소켓을 사용 , 하지만 클라이언트에서도 그럴 필요가 있을까? 라는 생각이 든다. 우선 connect가 잘 되는지 확인 해야 하기 때문에 실습으로 연습 하였던 코드를 조금 수정하여 네트워크 클래스를 설계하였다.  -  - 클라이언트 부분에서 send와 recv를 따로 만들어 두어서 클라이언트의 키입력을 할 때마다 편하게 호출해서 사용할 수 있게 만들었다.  -  -  - 실행 결과 여러 개의 클라이언트가 커넥트 되어 서버 내부에 있는 array<PLAYER>컨테이너에 ID를 부여받고 저장되는 것 까지 확인되었고 접속이 끊겼을 때 위 사진처럼 에러를 뱉어내며 컨테이너에서 클라이언트를 제거하는 모습까지 확인 되었다.  - 여기서 동기화를 시키는 과정에서 잡다한 모델들과 불필요한 부분이 많다고 생각이 들어 동기화를 위한 플레이어와 터레인만 깔려있는 클라이언트를 따로 요청하여 추후 작업을 진행  서버 프레임 워크  - 클라이언트와 Connect를 확인하고 무사히 recv를 받는 것 까지 확인했으나 기존에 내가 만든 서버 프레임워크가 너무 맘에 들지 않는다. 따라서 조금 더 깔끔하고 메모리를 효율적으로 관리할 수 있게 쓰레드 매니저를 생성하고 편의성을 위해 몇가지 환경설정을 바꾸며 새로운 서버 프레임워크를 제작 해보기로 결정하였다.  - 새로운 브랜치를 생성하여 서버 프레임워크 설계를 진행해보기로 했다. “인프런”이라는 사이트에서 “게임서버” 강의를 토대로 필요한것만 뽑아서 사용하기로 하엿다.  - ServerCore를 라이브러리로 생성하여 네트워크에 필요한 것들을 클라 서버 양쪽에서 다 사용할 수 있도록 만들었다.  -  - Types.h  -  - 위처럼 기존에 표준 자료형들을 사용하기 편하게 매핑하여 두었다.  - ThreadManager  -  - 그냥 쓰레드를 사용해줘도 되지만 TLS(Thread Local Storage)를 사용하는 경우가 발생할 수 있기 때문에 그 메모리를 묶어서 관리하면 좋을 것 같아 쓰레드매니저를 생성하게 되었다. 위에는 실제 제작 부분이며 Launch와 Join을 통해 쓰레드의 생성과 실행을 관여한다.  -  - Launch에서 쓰레드를 LockGuard로 소유권을 보장해주며 쓰레드의 저장공간에서 쓰레드함수를 실행시키게 했다.  -  - 서버쪽에서 이런식으로 사용하면  -  - 쓰레드가 잘 생성되는 것을 확인 할 수 있다.  패킷 정하기  - 패킷을 아직 제대로 설계하지 못하였다. 4주차에 동기화를 실행 시키면서 패킷설계를 하하는 것 낫다고 판단  C++  - 연산자 오버로딩  - Array vector map  유니티 맵 리소스 수집   1. 정경서(클라이언트)   **애니메이션 추출 관련 오류 해결**   * 예상 원인  1. 애니메이션 빼고 추출한 경우   이 부분 오류를 확인하기 위해 각 랩프로젝트의 모델 추출 코드를 비교하여 798 버전의 추출 코드를 사용하기로 했다. 또한 유니티에서 애니메이션을 추출할 때 다른 오류를 방지하기 위해 idle 애니메이션 하나만 선택하여 추출하였다.   1. 애니메이션 재생 번호 입력 오류의 경우   애니메이션 재생 함수에 번호를 넣을 때 오류가 발생하지 않도록 1개의 애니메이션만 추가하여 추출했으니 (0,0)으로 입력해주면 된다.   1. 애니메이션 함수 호출을 하지 않은 경우   해당 부분을 확인하기 위해 다른 애니메이션을 모두 지우고 하나의 애니메이션만 추출하였다.   1. 뼈 행렬의 이름이 다른 경우   교수님 코드는 휴머노이드 기준으로 추출한다고 배웠기 때문에 뼈의 이름이나 구조에 대한 의심을 해보았다. 하지만 현재 애니메이션이 작동하는 다른 모델은 Rig에서 Animation type이 Generic으로 설정 되어있어 해당 부분이 애니메이션 작동과는 관련 없다고 판단하였다. 이후 여러 방법을 통해 뼈 행렬을 변경하려고 시도하였으나 모델에 문제가 생길수 있다고 판단하여 이용희 교수님께 질문한 이후 진행하려고 한다.  -> 뼈의 이름은 그냥 변경해도 상관없다. 하지만 뼈의 이름은 유니크 해야한다는 것에 유의하며 이름을 변경하자.  위에 서술한 모든 방법을 다 적용해보았으나 해결되지 않았다. 이용희 교수님은 유니티에서 좌표계를 0,0,0으로 맞추어 보라고 하셨지만 이미 0,0,0으로 맞추고 추출하는 중이었다.  졸업작품을 진행한 조창근 선배님에게 질문해보니 이미 답을 알고 계셨다. 같은 모델러에게 에셋을 구매하여서 같은 문제를 겪은적이 있다고 하셨다. 먼저 힌트만 주셨는데 뼈 이름이 겹치는 것을 생각해보라고 하셨다. 이를 통해 뼈의 이름 앞부분이 겹치는 것을 모두 수정하였다.  하지만 가장 간단하게 해결하는 것은 터레인을 평지로 만드는 방법이었다. 졸업작품을 진행할 때 터레인을 사용하지 않는다면 해당 문제는 고민할 필요가 없는 문제인 것 같다.  터레인을 모두 평지로 만들고 나서도 문제가 발생하였는데 모두 x축을 기준으로 30도정도 회전되어 있다. 애니메이션 추출하는 것에 이번주를 다 소비하여서 해당 문제는 다음주에 해결해야 할 듯하다.  각 모델에 맞는 애니메이션을 txt, bin 파일로 추출하고 각 모델의 애니메이션 정보를 노션에 기록해 두었다.  **서버 동기화 목적 프로젝트 제작**   1. 평지일 것 2. 플레이어가 하나 있을 것   이 두가지 조건을 만족하는 프레임 씬 하나를 제작했다. Labproject07-8-9를 기반으로 하여 제작하였다.   1. 홍예나(클라이언트)   - 애니메이션 블렌딩 공부  - C++ 상속 추가로 공부  1. C++ 상속 추가로 공부  <업 캐스팅과 다운 캐스팅>  Class a : public b  클래스 a는 b의 모든 기능을 포함함  클래스 a는 b의 기능을 수행할 수 있기 때문에 a를 b라고 칭해도 무방함  즉, 모든 a는 b라는 것을 의미  파생될수록 특수화, 기반 클래스로 거슬러 올라갈수록 일반화 된다고 생각  ~ 업캐스팅  : 파생 클래스에서 기반 클래스로 캐스팅 하는 것  class derived : public base → derived는 base와 같다.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  #include <iostream>  #include <string>  class Base {  std::string s;  public:  Base() : s("기반") { std::cout << "기반 클래스" << std::endl; }  void what() { std::cout << s << std::endl; }  };  class Derived : public Base {  std::string s;  public:  Derived() : s("파생"), Base() { std::cout << "파생 클래스" << std::endl; }  void what() { std::cout << s << std::endl; }  };  int main() {  Base p;  Derived c;  std::cout << "=== 포인터 버전 ===" << std::endl;  Base\* p\_c = &c;  p\_c->what();  return 0;  }  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Base\* p\_c = &c;  이 코드처럼 base 객체를 가리키는 포인터가 derived 객체를 가리켜도 괜찮다  derived는 base를 상속받아서 derived == base라고 볼 수 있기 때문  ⇒ derived에는 base의 모든 정보가 들어있음  p\_c->what();  그러므로 위와 같은 코드 실행 시 “기반”이 출력됨  ~ 다운 캐스팅  업 캐스팅과 반대되는 개념  base에는 derived의 정보 중 없는 것이 있을 수 있기 때문에 되도록이면 하지 않는 것이 좋음  강제로 다운 캐스팅을 하기 위해서는  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Base p;  Derived c;  std::cout << "=== 포인터 버전 ===" << std::endl;  Base\* p\_p = &c;  Derived\* p\_c = static\_cast<Derived\*>(p\_p);  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  위와 같이 static\_cast로 강제로 다운 캐스팅 해야 함  Derived\* p\_c = dyanmic\_cast<Derived\*>(p\_p); → C++ 에서는 상속 관계에 있는 두 포인터들 간에 캐스팅을 해주는 dynamic\_cast 라는 것을 지원  → 근데 이걸로 컴파일하면 오류 뜸  <상속과 가상함수 virtual>  ~ 가상함수(virtual)  → 가상함수는 부모 클래스에서 상속받을 클래스에서 재정의할 것으로 기대하고 정의해놓은 함수  → 한번 가상함수로 선언된 함수는 따로 virtual 키워드를 앞에 써주지 않아도 가상함수로 인식함  \*override → 기반 클래스의 virtual 함수를 파생 클래스에서도 사용할 때 앞에 override를 붙이면 함수를 잘못 써서 오버라이드 되지 않는 오류를 쉽게 찾아낼 수 있다.  2. 애니메이션 블렌딩  애니메이션 블렌딩이란 애니메이션과 애니메이션의 사이에 둘의 중간 단계의 애니메이션을 삽입하여 자연스럽게 변화하도록 해주는 것  이를 구현하기 위해서 그 중간 단계인 두 가지 애니메이션이 섞이는 것을 먼저 구현해 봄    적 객체 중 하나의 애니메이션을 달리기 + 점프 두 가지의 가중치를 다르게 하여 섞어서 실행되도록 구현함  다리가 달리는 애니메이션에서처럼 앞, 뒤로 움직이면서 위 아래로 점프함    이 애니메이션을 키를 눌러서 애니메이션이 바뀌는 순간 한 번 실행되도록 구현해야 하는데 잘 되질 않아서 아직 완성하지 못함    현재 점프하는 애니메이션의 포지션 값이 0.3 이하일 때 두 가지 애니메이션이 다른 가중치로 섞여서 실행되도록 하려고 하는데 이 두 가지 애니메이션이 동시에 활성화되는 순간 렌더링이 되지 않는 오류가 있음 | | | |
| **다음주 할 일** | | | |
| 1. 신동엽(서버)   - 서버 프레임워크 설계 Lock, Session  - C++ 공부: 캐스팅, 함수 포인터, 템플릿  - 유니티 우주선 지역 맵 제작   1. 정경서(클라이언트)  * C++ 공부: 상속과 가상에 대한 공부 * 프레임워크 수정 * 애니메이션 관련 오류 수정  1. 홍예나(클라이언트)  * 애니메이션 블렌딩 구현, * stl 컨테이너 공부 * 상속 추가 공부 | | | |
| **비고** | | | |
|  | | | |