|  |  |  |  |
| --- | --- | --- | --- |
| [Cranding(크랜딩)] | | | |
| **주차** | **2023.12.24 ~ 2023.12.30** | **작성자** | **정경서** |
| **주간 회의** | | | |
| **회의 날짜** | | 2023.12.27 | |
| 1. Notion 기록하기   * 매일 본인 공부 기록 작성하기 * 마감 시간: 23시 59분 * 금요일 10시에 주간 보고서 작성하기   2. git repository 생성하기   * main 브랜치 → 최신 릴리즈 버전만 * 각자 브랜치 만들어서 작업 후 병합하기 * ignore 파일 하나로 통일하기   3. 방학 계획 세우기  - 금요일까지 각자 주차 별 계획 세워오기.  4. 각자 씹어먹는 C++ 공부하기  - Notion 공부 기록에 추가하기 | | | |
| **이번주 한 일** | | | |
| 1. 신동엽(서버)   1주차  C++ 복습  (서버 프레임 설계 전 C++개념에 대해 한번 다시 정리하고 하는 것이 좋을 것 같아서)   * **포인터와 레퍼런스 ( 참조 )** * **정리 내용** * 포인터: 일반적인 자료형 int num같은 것들을 int 자료형을 담는 바구니로 비유하자면 포인터는 주소를 담는 바구니라고 생각하면 된다. * 왜 사용하는가? 메모리의 유효범위 때문에 사용한다. * 포인터는 8바이트로 크기고정 그렇다면 자료형을 붙이는 이유는? 무엇이 담겨있는지 유추할 수 있다. 또한 포인터를 이용해서 데이터를 조작할 수 있다 보니 데이터가 덮어써지는 문제가 발생할 수 있다. 강제로 캐스팅해서 넣는 상황을 만들지 말자. * 참조형: 로우레벨 단계에서 포인터와 작동방식은 완전히 일치한다. * C++관점에서 예를 들어 number라는 변수가 있다면 이 변수의 이름에 또 다른 이름을 부여하는 것 * 그냥 값 전달하는 것처럼 사용이 가능하며, 주소값을 이용해 그 안에 있는 데이터를 직접 건드리는 방식이다. * 포인터 VS 참조 (ref) * 가독성: 포인터는 주소를 넘기니까 확실하게 원본을 전달해준다는 힌트를 줄 수 있다. 참조는 일반 값전달과 방식이 같기 때문에 모르고 지나치는 경우가 발생할 수도 있다. * ref앞에 const를 붙여 마음대로 데이터를 개선하는 부분을 막을 수 있다. * 초기화: 참조 타입은 앞서 별칭이라고 하였는데, 따라서 참조하는 대상이 없으면 안 된다. 하지만 포인터는 초기값이 null 또는 0으로 설정되어도 괜찮다. 어떤 ~ 주소 라는 의미이기 때문에 실존하지 않을 수 있다. 또한 포인터 연산이 가능하다. ++ -- 등. * null값을 받을 수 있는 포인터는 실제 게임 개발 시 아이템 찾기, 몬스터 찾기 등 find에서 사용할 수 있다. * 결론   + 무엇이 우월한지에 대해선 정해진 답이 존재하지 않다.   + 값이 없을 수도 있는 경우를 고려해야 한다면 포인터를 사용하자 ex) find   + 바뀌지 않고 오직 값을 읽는 용도로 사용한다면 const를 붙여 ref를 사용하자   + 그 외에 편의성이 ref가 좋기에 명시적으로 앞에 OUT을 붙여서 ref를 사용하자   **게임서버 프로그래밍 텀프로젝트 복습**   * 패킷 재조립   + 게임서버 프로그래밍 수업 때 배운 패킷 재조립 방식을 그대로 사용.     - 나만의 재조립 방식을 사용해봐야 함. * Mutex 문제점 발견   + 무분별한 lock을 사용하여 성능 저하     - Lock의 사용을 최소화   방학 목표: 서버 프레임워크 제작 완료 + 사용될 맵 제작 완료 + 클라이언트 동기화  1주차: C++ (포인터, 참조) + 게임서버프로그래밍 텀프로젝트 미흡한 부분 개선  2주차: C++ 상속 다형성 + 게임서버프로그래밍 강의 복습(atomic, session, mutex) + 서버 프레임워크 제작 +패킷 정리  3주차: STL(array, vector, map) + 게임서버프로그래밍 강의 복습(thread manager) + 서버 프레임워크 제작+패킷 정리  4주차: 게임서버프로그래밍 강의 복습(send, recv buf) + 서버 프레임워크 제작+패킷 정리  5주차: 게임서버프로그래밍 강의 복습(DB연동) + 서버 프레임워크 제작 + 맵 제작  6주차: 서버 프레임워크 제작 + 맵 제작  7주차: 서버 프레임워크 제작 + 맵 제작  8주차: 서버 프레임워크 제작 + 맵 제작  9주차: 서버 프레임워크 제작 + 동기화  10주차: 서버, 클라이언트 동기화   1. 정경서(클라이언트)   1주차 공부     * 유니티를 통해 애니메이션을 적용하고 해당 모델과 애니메이션의 내용을 추출하여 적용하는 과정을 공부함. * 애니메이션 컨트롤러 및 클립 관련 실습 진행 * 애니메이션 관련 샘플 코드 분석 * 프레임워크 수정을 위해 게임 프로그래밍 패턴이라는 책을 읽었습니다   방학 목표: 클라이언트 프레임워크 제작 완료 + 애니메이션 구현 완료 + 디퍼드 렌더링 70% 이상 구현  1주차: 애니메이션 공부  2주차: 애니메이션 공부 + 프레임워크 수정  3주차: 애니메이션 구현 + 프레임워크 수정  4주차: 애니메이션 구현 + 프레임워크 수정  5주차: 애니메이션 구현 + 프레임워크 수정  6주차: 애니메이션 구현 + 프레임워크 수정 + 디퍼드 렌더링 공부  7주차: 애니메이션 구현 + 프레임워크 수정 + 디퍼드 렌더링 구현  8주차: 애니메이션 구현 + 프레임워크 수정 + 디퍼드 렌더링 구현  9주차: 애니메이션 구현 + 프레임워크 수정 + 디퍼드 렌더링 구현  10주차: 애니메이션 구현 + 프레임워크 수정 + 디퍼드 렌더링 구현   1. 홍예나(클라이언트)   1주차   * 다렉 방학특강 애니메이션 관련 프로젝트 7-9-5      * cs코드 출력 내용과 cpp에서의 입력 내용이 일치하도록 구조 변경 * 애니메이션 컨트롤러, 애니메이션 클립 공부      * 전반적인 애니메이션의 개념 강의 수강   목표: 프레임워크 수정 + 애니메이션 구현 + 가우시안 블러 시작  - 1주: 애니메이션 공부  - 2주: 애니메이션 공부 + 프레임워크 수정  - 3주: 애니메이션 공부 + 프레임워크 수정  - 4주: 애니메이션 구현 시작 + 프레임워크 수정  - 5주: 애니메이션 구현 + 프레임워크 수정  - 6주: 여행 + 프레임워크 수정  - 7주: 프레임워크 수정 + 애니메이션 구현 + 가우시안 블러 공부  - 8주 프레임워크 수정 + 애니메이션 구현 + 가우시안 블러 구현 시작  - 9주: 프레임워크 수정 + 애니메이션 구현 + 가우시안 블러 구현  - 10주: 프레임워크 수정 마무리 + 애니메이션 구현 마무리 + 가우시안 블러 구현 | | | |
| **다음주 할 일** | | | |
| 1. 신동엽(서버)   - 서버 프레임워크 작업 시작  - 패킷, Network.cpp  - 우선 클라이언트와 연결 후 에코서버까지 (더미클라를 활용)  - C++ 상속, 다형성 개념 정리  - 게임서버 강의 복습: atomic, mutex, Session  - 클라이언트 코드 분석 이후 송수신 클래스 제작   1. 정경서(클라이언트) 2. 홍예나(클라이언트)  * 애니메이션이 키에 따라 바뀌도록 구현 * 게임 디자인 패턴 공부 * 다렉2 과제에 새로운 에셋 넣어보기 * C++ 상속 부분 공부 | | | |
| **비고** | | | |
|  | | | |