**게임서버 프로그래밍**

**텀프로젝트**

**게임공학부**

**엔터테인먼트컴퓨팅과**

**2016184041**

**차지환**

**프로그램 실행 환경 및 실행 방법.**

**실행환경**

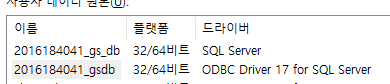
Cpu : Ryzen 3700x ( 8 core 16thread )

Ram : 32기가

Windwos Pro 64bit.

**DB 환경.**

**DSN : 2016184041\_gsdb**

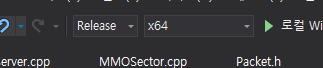


수업에서 사용한 ‘SQL Server’ 드라이버 사용하여 연결 시 테스트는 통과하는데 실제 서버에서 접속 시 time out이 발생하여 ODBC Driver 17 for SQL Server을 사용하고 있습니다.

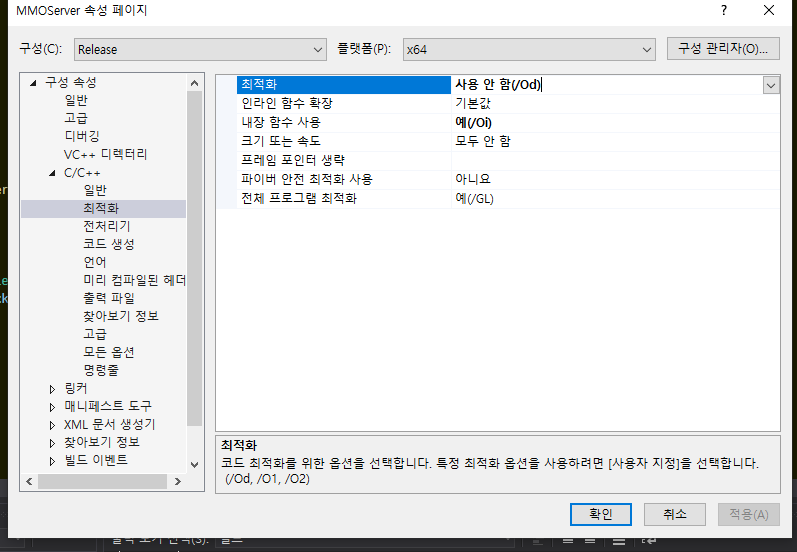
**필수 Visual Studio 2019 환경.**

**필수!!**

**Release Mode x64**

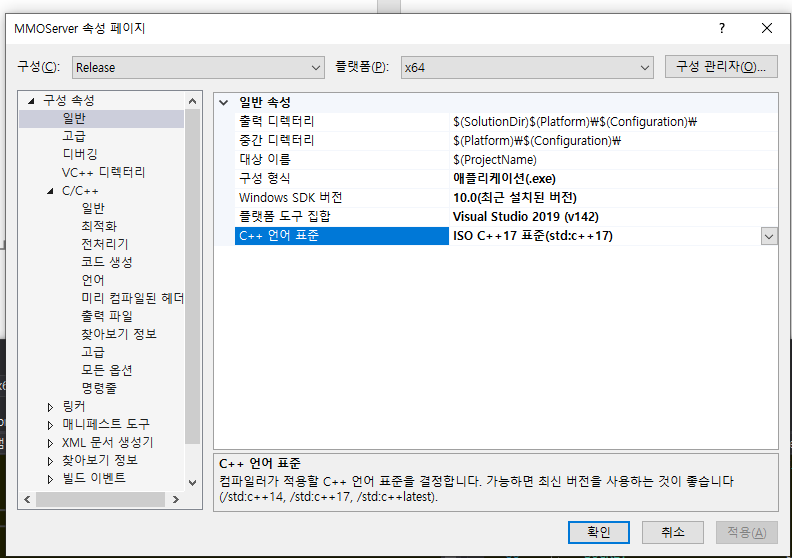
**.**

**구성 속성 -> C / C++ 최적화 -> 최적화 사용안함.**

****

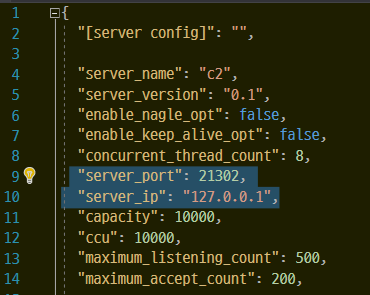
**필수 : 컴파일러 최적화를 꺼주시기 바랍니다.**

**구성 속성 -> 일반 -> C++ 언어 표준 c++ 17 표준.**

****

**서버 환경.**

**프로젝트 파일 내의 config.json 사용시 서버 설정 변경 가능.**

****

**현재 IP : 127.0.0.1 ( 루프백 주소 )**

**포트 : 21302**

****

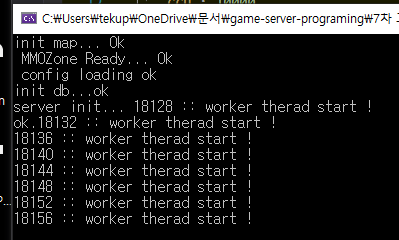
**Npc 숫자.**

****

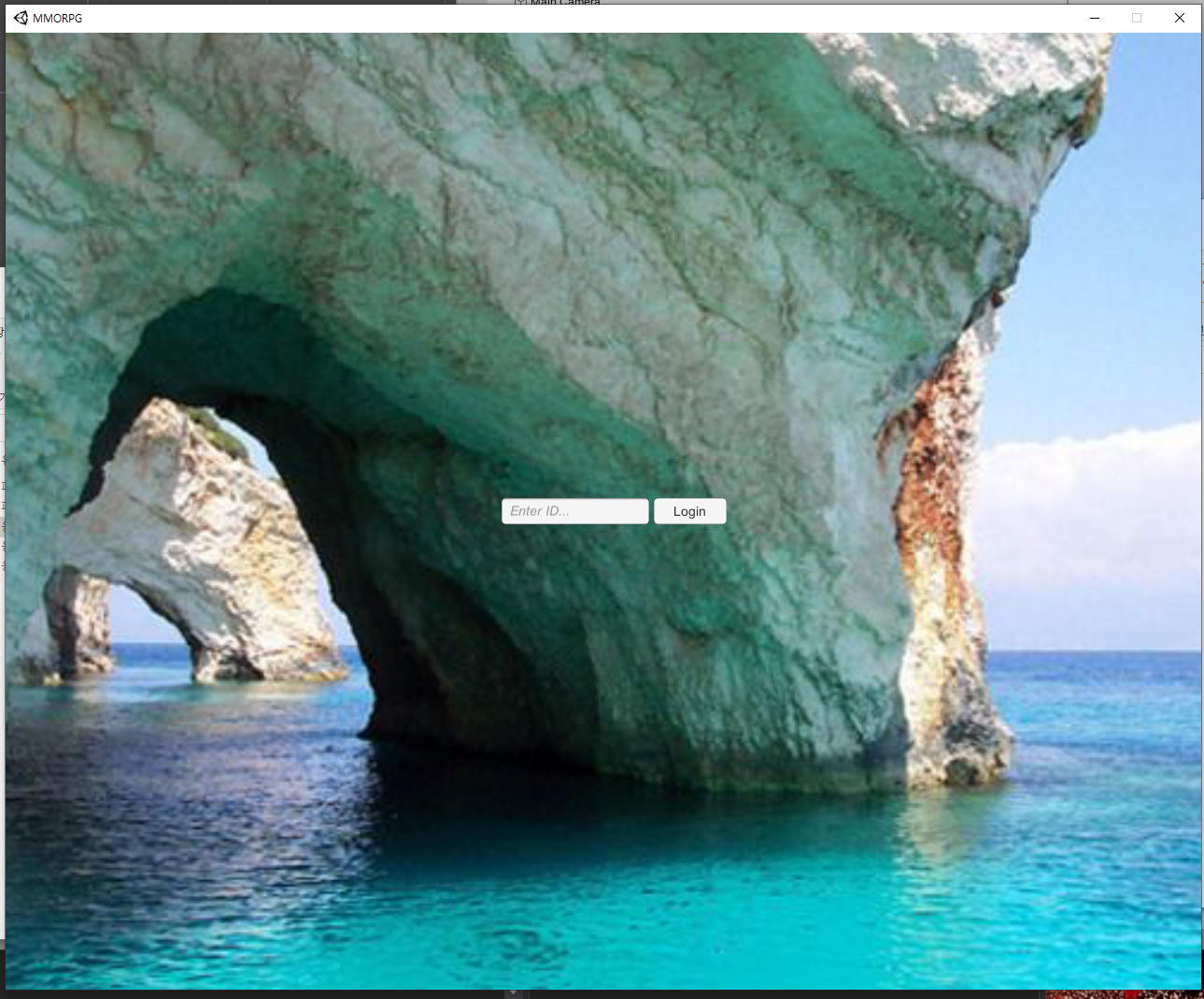
**워커 스레드 카운트.**

**실행 방법.**

서버를 실행.

****

**유니티 클라이언트 실행.**

****

****

**ID 입력.**

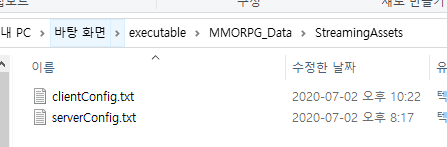
**DB에 없으면 -> 생성.**

**DB에 있으면 -> 정보를 가져와서 실행.**

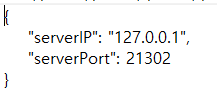
****

**실행 된다.**

**서버 IP또는 포트가 바뀌는 경우.**

****

**유니티 클라이언트 파일의 serverConfig.txt 열기.**

****

**IP와 Port를 변경하면 된다.**

**못 읽어오는 경우 txt 파일이 인코딩방식이 utf8(BOM 코드 없음)인지 확인.**

**서버 구조.**

**서버 부분.**

서버를 역할 분담할 정도로 부하가 크지 않아서 서버를 단일서버로 구현되어 있다.

**DB**

DB의 경우 읽기 작업은 여러 개의 스레드로 동기식으로 진행하며 작업이 끝나면PostCompletionQueueStatus()함수를 통해서 WorkerThread로 넘겨준다.

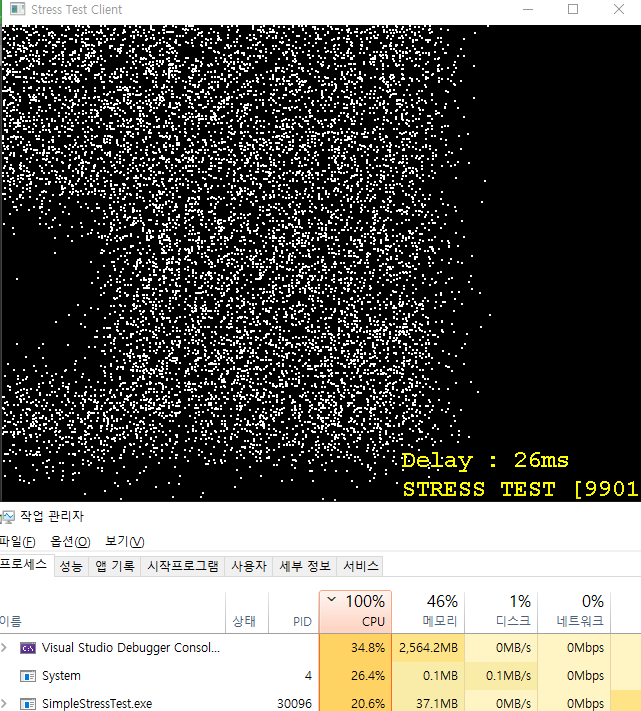
쓰기 작업은 단일 스레드에 MPSC큐로 작업을 넣어주어 하나씩 처리되도록 하고있다.

**콘텐츠 부분.**

검색 효율을 위해서 시야의 두배 크기의 섹터 단위로 분리.

타이머 스레드 스레드간 통신을 하지 않고 각 스레드 마다 타이머 작업을 처리한다..

**스트레스 테스트 결과.**





동접자 수가 늘어남에 따라 서버 지연시간이 상승은 했지만 플레이에 크게 무리가 가는 수준은 아니다.

실행 영상.

<https://www.youtube.com/watch?v=4bGojfB7upk&feature=youtu.be>