

게임서버 포트폴리오

김상윤

인적사항

김상윤

2001. 06. 17

grinder0617@gmail.com

010-7739-0617

경기도 시흥시 거주

2026.02 : 한국공학대학교 게임공학과 졸업예정

2020.03 : 한국산업기술대학교(현 한국공학대학교) 게임공학과 입학

2020.02 : 경구고등학교 졸업

프로젝트 소개

https://github.com/kimusamu/Game_Server_Project



SFML(Client)와 IOCP(Server)
를 이용한 2D MMORPG
게임 프로젝트로
Client와 Server 모두 제작

사용한 기술스택

언어 / 그래픽 프레임워크

C++, SFML

네트워크

Windows IOCP : 비동기 소켓 서버

데이터베이스

MSSQL : ODBC 연동

스크립팅

Lua : NPC AI 및 이벤트 처리

구현 목록

몬스터

20만마리, 2종류(랜덤 이동, A* 알고리즘 기반 추적 이동)
배치 좌표는 Script를 통해 결정

플레이어

Level, HP, Name, Attack, Inventory(HP Potion, Exp Potion, Gold)
자동 회복(10초가 지날 때 마다 HP 1 씩 회복)
죽을 시 10초 뒤 부활(최소 HP, Exp는 0으로 초기화)

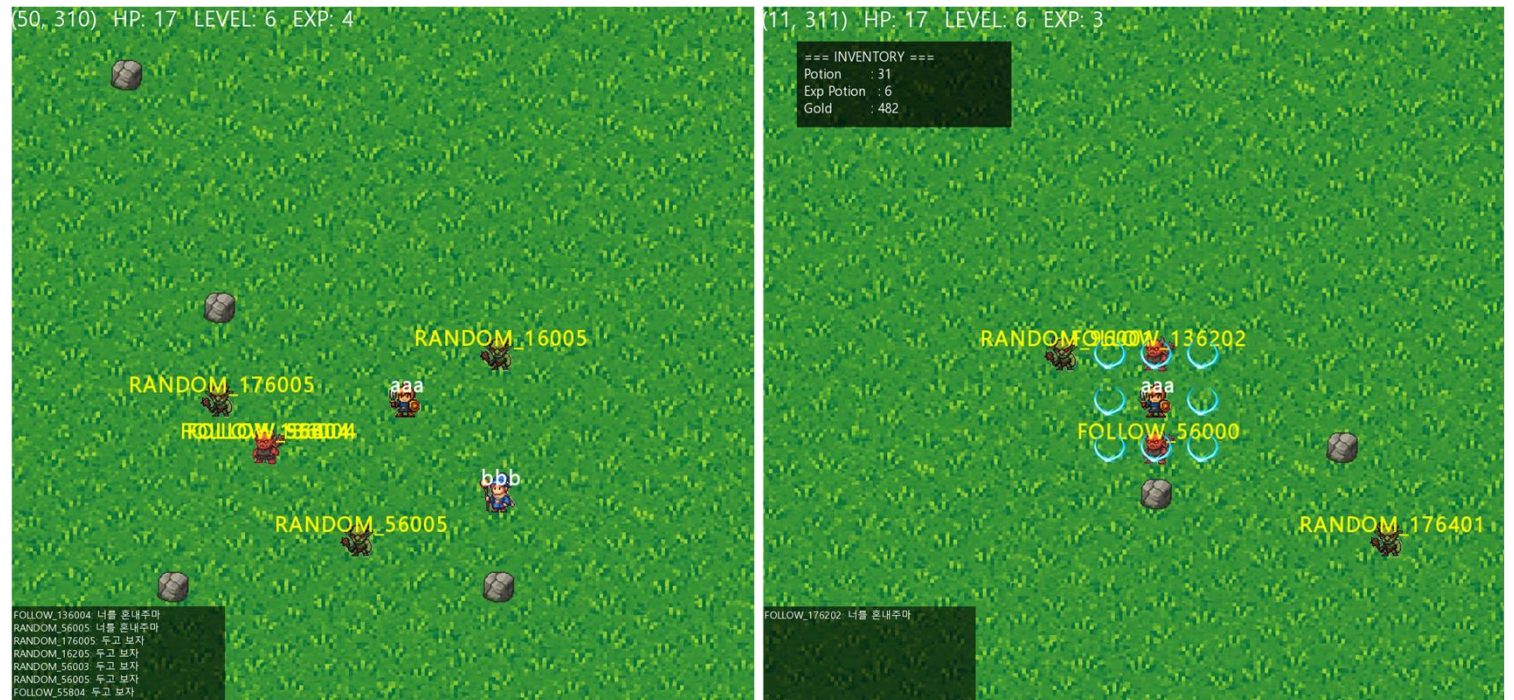
채팅 시스템

몬스터 -> 플레이어는 플레이어 자신한테만 보이게 표시
플레이어 <-> 플레이어는 채팅을 통한 양방향 소통 가능(한, 영 변환 가능)

데이터베이스

30초 주기로 DB에 자동 저장, 로그아웃 시 필수 저장
똑같은 ID가 동시 접속 시, 후에 접속 시도한 클라이언트 강제 종료
데이터베이스에 작성한 ID가 없을 시, 자동으로 ID 생성 후 게임 플레이

게임 플레이



방향키 : 이동 / I : 인벤토리 / Enter : 채팅 / Space : 공격
1 : 회복 포션 / 2 : 경험치 포션

https://www.youtube.com/watch?v=jB-pDFF_oCA

서버 생성

```
DSN=2025_GameServer CONNECT COMPLETE
OBSTACLES SPAWN COMPLETE !
MONSTER SPAWN START !
MONSTER SPAWN : ID = 10000, string="MONSTER10000" (ret=12)
MONSTER SPAWN : ID = 20000, string="MONSTER20000" (ret=12)
MONSTER SPAWN : ID = 30000, string="MONSTER30000" (ret=12)
MONSTER SPAWN : ID = 40000, string="MONSTER40000" (ret=12)
MONSTER SPAWN : ID = 50000, string="MONSTER50000" (ret=12)
MONSTER SPAWN : ID = 60000, string="MONSTER60000" (ret=12)
MONSTER SPAWN : ID = 70000, string="MONSTER70000" (ret=12)
MONSTER SPAWN : ID = 80000, string="MONSTER80000" (ret=12)
MONSTER SPAWN : ID = 90000, string="MONSTER90000" (ret=12)
MONSTER SPAWN : ID = 100000, string="MONSTER100000" (ret=13)
MONSTER SPAWN : ID = 110000, string="MONSTER110000" (ret=13)
MONSTER SPAWN : ID = 120000, string="MONSTER120000" (ret=13)
MONSTER SPAWN : ID = 130000, string="MONSTER130000" (ret=13)
MONSTER SPAWN : ID = 140000, string="MONSTER140000" (ret=13)
MONSTER SPAWN : ID = 150000, string="MONSTER150000" (ret=13)
MONSTER SPAWN : ID = 160000, string="MONSTER160000" (ret=13)
MONSTER SPAWN : ID = 170000, string="MONSTER170000" (ret=13)
MONSTER SPAWN : ID = 180000, string="MONSTER180000" (ret=13)
MONSTER SPAWN : ID = 190000, string="MONSTER190000" (ret=13)
MONSTER SPAWN : ID = 200000, string="MONSTER200000" (ret=13)
MONSTER SPAWN END !
```

```
math.randomseed(os.time())

function init_npc_pos(id)
    local idx = id - MAX_USER
    local cols = math.floor(W_WIDTH / 10)
    local baseX = (idx % cols) * 10 + 5
    local baseY = math.floor(idx / cols) * 10 + 5

    local jitter = 8
    local x = (baseX + math.random(-jitter, jitter)) % W_WIDTH
    local y = (baseY + math.random(-jitter, jitter)) % W_HEIGHT

    return x, y
end
```

데이터베이스 연결

장애물 생성

그 이후 몬스터 초기화 및 생성(1만 단위로 생성 되었는지 표시)

몬스터의 생성 좌표는 스크립트가 계산 후 위치를 받아 생성

몬스터 전부 생성 완성 시, 서버 생성 완료

클라이언트 로그인

=== GAME LOGIN ===

Server IP:

127.0.0.1

User ID (Max 10 chars):

```
playersS|
```

Enter Server IP first, then press TAB to switch to ID input.
Press ENTER to connect when both fields are filled.

Server IP와 User ID 입력 후
게임 접속

게임의 데이터베이스와 비교하여
존재하는 User ID 시
해당 User ID의 데이터를 가지고 와
게임 실행

존재하지 않을 시
ID 생성 후 초기 상태로 게임 시작

[illegible]

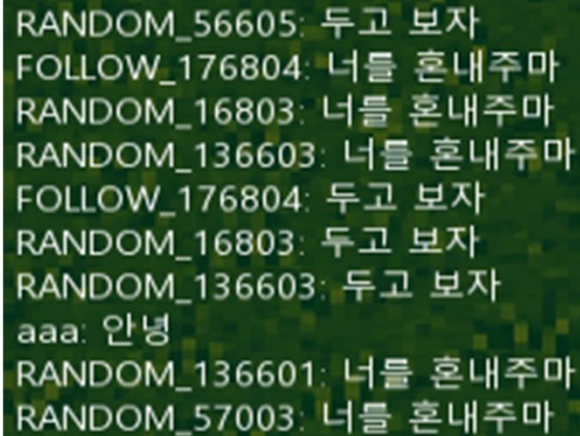
인벤토리



	user_id	user_x	user_y	user_hp	user_exp	user_level	user_potion	user_exppotion	user_gold
↖	adafgSS	0	0	3	0	1	0	0	0
	aaa	52	311	18	0	6	30	4	462
	ababab	1410	147	3	0	1	0	0	0
	admin	0	0	3	0	1	0	0	0
	bbb	47	311	2	5	1	4	0	46
	ccc	55	305	2	1	1	0	1	6
	dasdsadsa	715	1133	3	0	1	0	0	0
	ddd	50	50	3	0	1	0	0	0
	dsadas	1175	1302	3	0	1	0	0	0
	iog	875	1804	3	0	1	0	0	0
	kimusamu	1264	1587	4	3	2	2	0	31
	yaho	29	33	3	0	1	0	0	0
	zzz	8	11	3	0	1	0	0	0
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

I키를 입력 시 인벤토리 창이 나오며, 체력 포션, 경험치 포션, 골드 표시
체력 포션, 경험치 포션은 몬스터 처치 시 랜덤 확률 획득
골드는 1-10 중 랜덤한 골드량 획득
또한 데이터베이스에 실시간 저장으로 접속 종료 시 분실 우려 없음


채팅창



RANDOM_56605: 두고 보자
FOLLOW_176804: 너를 혼내주마
RANDOM_16803: 너를 혼내주마
RANDOM_136603: 너를 혼내주마
FOLLOW_176804: 두고 보자
RANDOM_16803: 두고 보자
RANDOM_136603: 두고 보자
aaa: 안녕
RANDOM_136601: 너를 혼내주마
RANDOM_57003: 너를 혼내주마

몬스터 반응 대사는
반응을 유도한 플레이어에게
나오도록 설정

플레이어가 작성한 채팅은
채팅을 작성한 플레이어가
존재하는 해당 섹터 내
플레이어 모두가 볼 수 있음



aaa: 안녕하세요
bbb: 반갑습니다

영어가 아닌 한글로도
소통이 가능함

서버 상태 확인

```
DB SAVE ...
DB_SavePlayerPosition COMPLETE
57803'S HP : 2
57803'S HP : 1
57803 IS DEAD.
DROPPED : EXP POTION
DROPPED GOLD : 1
DB_LoadPlayerPosition COMPLETE
[NPC 57803] RESPAWN AT (31, 392).
DB SAVE ...
DB_SavePlayerPosition COMPLETE
DB SAVE ...
DB_SavePlayerPosition COMPLETE
[Player 0] HIT. HP: 11
[NPC 137804] HIT. HP: 2
[Player 0] HIT. HP: 10
[NPC 178206] HIT. HP: 2
[Player 0] HIT. HP: 9
[NPC 96805] HIT. HP: 2
aaa HEAL UP
aaa HEAL UP
SEND CHAT
SEND CHAT
DB SAVE ...
DB_SavePlayerPosition COMPLETE
DB SAVE ...
DB_SavePlayerPosition COMPLETE
DB_SavePlayerPosition COMPLETE
DB_SavePlayerPosition COMPLETE
DB_SavePlayerPosition COMPLETE
|
```

서버에서 발생하는 사건들을
한 눈에 파악 가능

데이터 저장
아이템 획득
플레이어, 몬스터 상태
채팅 전송 등등

STRESS TEST



Stress Test 결과

최소 2950 Dummy Clients

최대 4150 Dummy Clients

동시 접속이 가능한 것을 확인 가능

몬스터

랜덤이동 10만마리,

A* 기반 추적이동 10만마리

CPU

Intel® Core™ i5-12500H

Memory

32GB

서버 최적화

IOCP 기반 고성능 I/O 파이프라인

의도 : 수천 동접에서 컨텍스트 스위치 / 락 경험 최소화 후 선형 확장성 확보

구현 : CreateIoCompletionPort로 소켓을 IOCP에 바인딩 후,
WorkerThread가 GetQueuedCompletionStatus 루프에서
OP_ACCEPT / OP_RECV / OP_SEND / OP_NPC_MOVE를
이벤트 기반으로 처리
새 연결은 AcceptEx 완료 이후 즉시 IOCP에 등록.

효과 : Blocking 대기 제거, N:1 이벤트 분배 -> Worker 수 만큼 확장

스트림 패킷 프레이밍 + 재조립(가변 길이 1byte 헤더)

의도 : 불완전 수신 / 과수신에서 불필요한 복사와 파싱 비용 최적화

구현 : p[o](길이)로 반복 파싱, 남은 byte는 prev_remain에 보관 후
버퍼 앞쪽으로 memcpy하여 재사용

효과 : 분할 수신 / 버스트 수신에도 힘 / 파싱 오버헤드 고정화

공간 분할(섹터링) + 가시 리스트 후보 축소

의도 : 전체 엔티티 탐색을 인접 섹터로 축소하여 CPU 및 lock 비용 절감

구현 : 좌표 -> 섹터 인덱스 해시

3 * 3 인접 섹터만 lock 후 후보 수집

상태 / HP / 거리로 필터링

효과 : 밀집 장면에서도 가시 연산량과 lock 홀드 시간이 크게 감소

서버 최적화

뷰 리스트 증분 동기화

의도 : 대역폭을 절약하면서 체감 반응성 유지

구현 : 이동시 old_vlist와 near_list 비교를 통해 Add / Move / Remove 전송
(Player -> Peer와 Peer -> Player 양방향)

NPC 또한 old_vl <-> new_vl 비교로 동일 패턴 적용

효과 : 패킷 수와 크기 감소, 혼잡 구간에서 Round-Trip Time 흔들림 완화

섹터 단위 장애물 스트리밍

의도 : 맵 전체 송신 대신 필요 섹터만 전송하여 네트워크 및 CPU 절감

구현 : 로그인 / 리스폰 / 요청 시 현재 섹터의 인접 섹터 타일만 송신
섹터가 바뀔 때는 빠진 섹터는 Remove + 새 섹터 Add 만 수행

효과 : 대규모 맵에서도 대역폭 사용이 섹터 면적에 고정

A* 경로 탐색의 메모리/캐시 최적화

의도 : 다수 NPC 동시 추적 시 Heap 할당 / 메모리 초기화로 인한 Jitter 제거

구현 : thread_local NodePool(고정 배열)로 노드 zero-allocation

방문 배열은 epoch 스탬프(overflow 시에만 전체 0으로 초기화)

맨해튼 휴리스틱 + Small Priority Queue 사용

효과 : per-tick A* 비용 안정화로 flame jitter 줄어듦, Core 확장성 늘어남

서버 최적화

타이머 큐 + IOCP 통합 스케줄링

의도 : 시간 기반 이벤트(랜덤 이동 / 리스폰 / 힐)를

별도 timer thread 폭증 없이 처리

구현 : 만기 이벤트만 꺼내 OP_NPC_MOVE를 IOCP에 포스트

리스폰 / 힐은 즉시 처리

주기 저장은 30초 간격으로 증분 수행

효과 : thread 재사용으로 캐시 지역성 유지, busy waiting 제거

락 경합 최소화

의도 : 전역 lock 없이 섹터 / 세션 수준으로 임계구역을 축소

구현 : 섹터 컨테이너는 섹터 별 mutex만 잠금

세션 상태는 필요한 구간만 s_lock 활용

효과 : 이동 / 가시 / 전투가 동시 다발적으로 일어나도 throughput 유지

중복 작업 억제

의도 : 같은 이벤트 / 계산의 재스케줄 및 재계산 방지로 timer / AI 병목 감소

구현 : NPC 부활 / 이동은 is_active / greet_moves_left로 상태 제어

필요 시에만 다음 이동 스케줄, 경로는 타깃 변화 / 소진 시에만 repass

효과 : tick spike 완화, 평균 지연 하락

서버 최적화

동시성 컨테이너 활용

의도 : 연결(Session) 전역 접근에서 중앙 lock 제거

구현 : `concurrent_unordered_map<int, shared_ptr<SESSION>>` clients 사용
 `concurrent_priority_queue<event_type>` timer_queue 사용

효과 : 다수 worker / timer가 동시에 접근하더라도 확장성 유지

프로젝트 최종 정리

1. 서버 초기화

WSAStartup, DB 연결

맵 장애물 스폰, NPC 초기화(Lua)

IOCP / WorkerThread + TimerThread 실행

2. 클라이언트 처리

AcceptEx로 접속 수락

Packet 처리(Login, Move, Chatting, Attack, Potion)

잘못된 접속 / 에러 시 Disconnect

3. 게임 로직

월드 / 시야 관리(섹터링, 가시범위)

전투 시스템(피격, 무적, 리스폰, 드랍)

NPC 이동 / 추적(랜덤, A* 기반 추적)

4. 스크립트 연동

Lua 스크립트로 NPCAI 동작 제어

API 제공(좌표, 대화 행동 제어)

5. 타이머 & 데이터베이스

TimerQueue : NPC 이동, 리스폰, 회복 이벤트

30초마다 DB 저장(위치 / 스탯 / 인벤토리 등)

감사합니다

김상윤

grinder0617@gmail.com