네트워크 게임 프로그래밍 Term Project

2019180016 백종화

2019182007 김문수

2019182009 김석희

**애플리케이션 기획 (무슨 교과목에서 누가 작업한 게임인지 명시할 것)**

김석희, 백종화 학우가 윈도우 프로그래밍에서 C++로만든 프로젝트를 이용할 예정입니다.

게임 설명

A. 게임 소개 및 특징

-> 적을 막는 디펜스 게임과 카드 장르를 합친 게임입니다. 또한 카드 게임 장르에서 덱 빌딩이라는 다양성으로 매 게임당 새로운 나만의 조합으로 즐길 수 있습니다.

-> 캐릭터가 적을 공격 할 수 있는 방식이 카드 밖에 없기에 적절한 카드 분배와 마나 관리같은 전략적 판단을 요구하기도 합니다. 또한 덱 빌딩 시 강력한 카드는 마나 소모가 크게 설정 되어 있어서 밸런스 있는 덱 빌딩을 중요로 합니다.

B. 게임 규모

-> 크게 총 3웨이브로 이루어져 있습니다. 웨이브에 따라 추가 되는 몬스터가 있으며 최종적으로 보스를 잡으면 끝납니다. 그 전에 자신의 타워가 피해를 입어 파괴되면 패배로 끝납니다. 몬스터는 지상 몬스터, 공중 몬스터가 있고, 보스까지 총 3종류가 존재합니다.

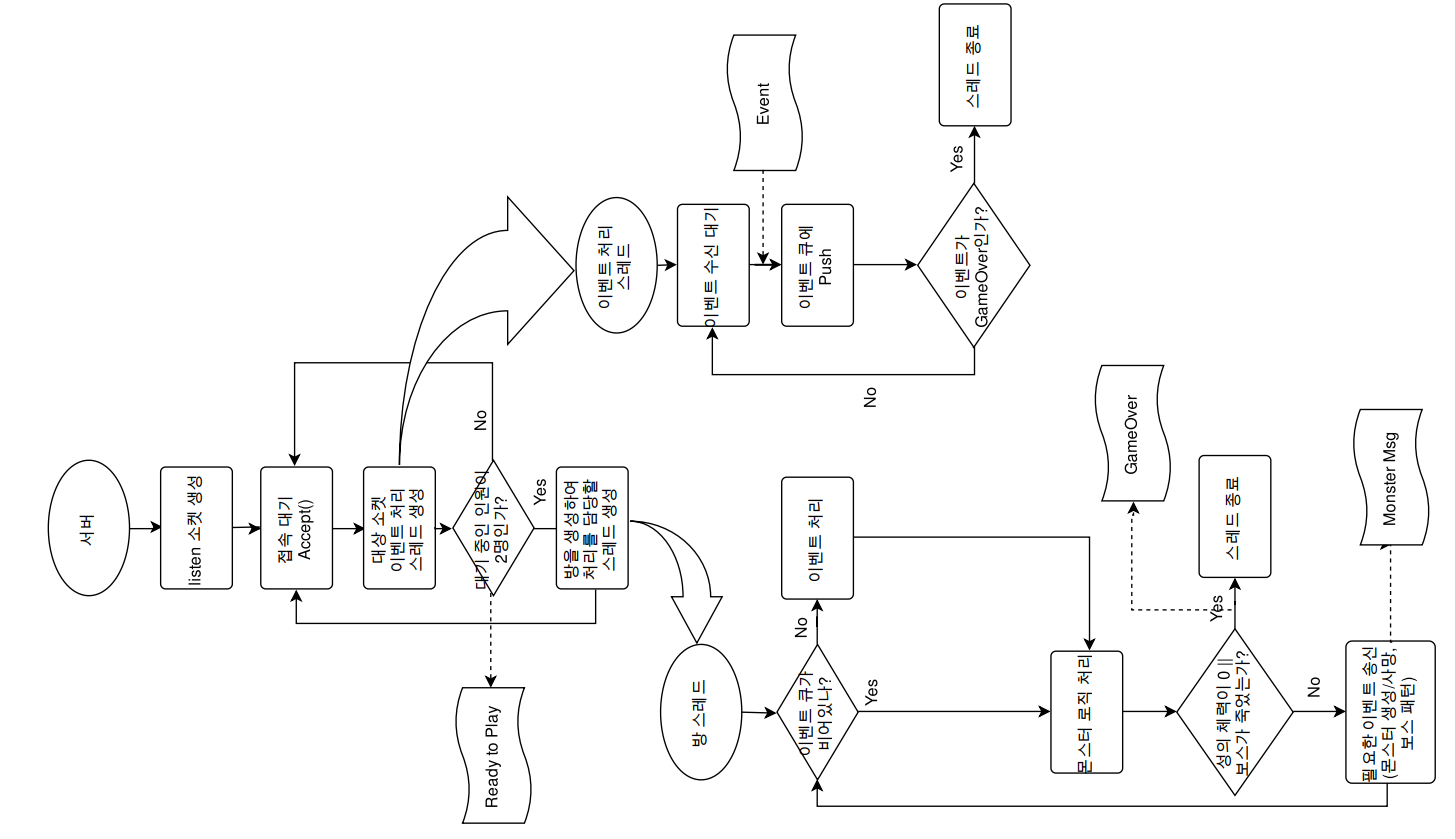
-> 게임의 메인이라고 할 수 있는 카드는 얼핏보면 많지 않아 보입니다. 베이스가 되는 스킬 카드는 10개 밖에 없기 때문입니다. 자주 쓸 수 없는 각성기 기술까지 하면 11개의 스킬이죠. 하지만 카드 하나하나에 트라이포드라는 시스템이 있어 카드를 강화하거나 다양하게 바꿀 수 있습니다. 3단계로 이루어져 1단계 3가지, 2단계 3가지, 3단계 2가지 중 하나씩 골라 총 18개의 경우의 수가 한 카드에서 나옵니다.

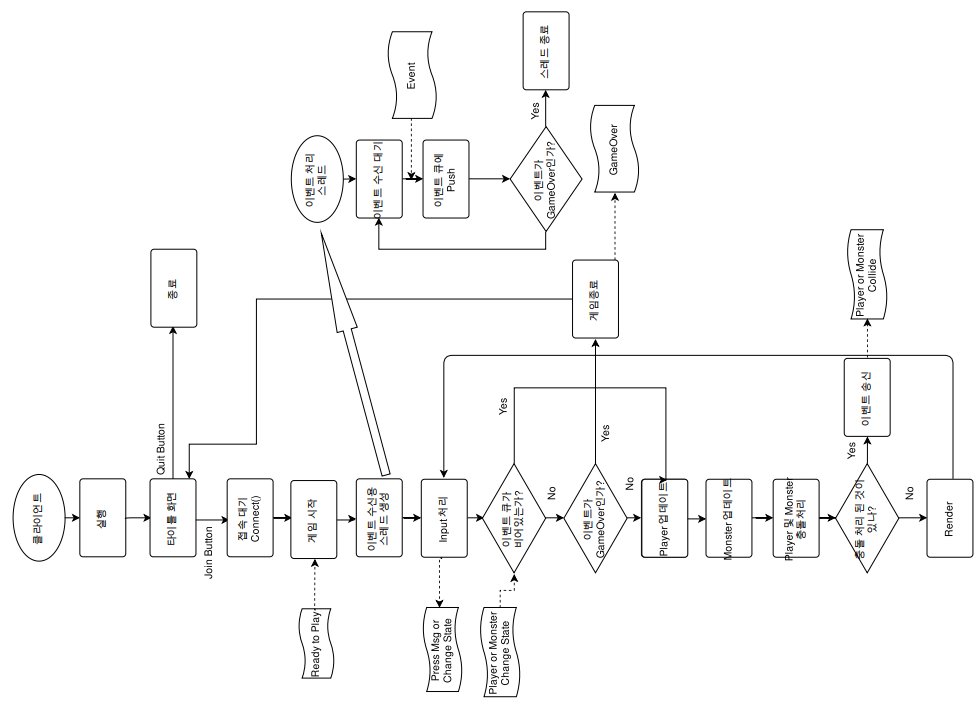
-> 골드와 상점 시스템이 존재하여 자신의 카드를 바꿀 수 있습니다. 몬스터를 잡으면 골드와 경험치가 증가합니다. 경험치를 얻어 레벨업하면 상점이 열려 모아둔 골드로 새롭거나 기존에 있던 카드를 덱에 추가할 수 있습니다. 그 뿐만 아니라 기존에 있던 카드 다음 단계의 트라이포드를 구매하여 강화시킬 수 있습니다.

실행방법

-> 키보드의 w, a, d키로 움직입니다. 마우스 드래그 앤 드랍으로 카드를 사용할 수 있으며, 일정 게이지를 다 채워야 쓸 수 있는 각성기는 E키로 쓸 수 있습니다.

**High-level 디자인**





**Low-level 디자인**

변수 작성 법: (멤버 변수라면: m\_)(자료형타입)(변수명) ex:m\_iNum

변수명은 파스칼 표기법으로 작성 ex: PersonState

함수명은 동사

이벤트 방식 정의:

사용할 열거자:

enum EventSender {Server, Client1, Client2};

enum EventType { Monster\_Spawn, Monster\_Hp, Player\_Move, Player\_Use\_Card, Castle\_Hp, … };

enum MonsterType {M\_Wolf, M\_Bat, M\_Boss}

enum CardType { … }

enum PlayerMove{ … }

사용할 변수:

BYTE EventDefine; // 이벤트를 정의하는 변수, 하위 6비트는 보내는 내용 상위2비트는 보내는 사람을 비트 연산으로 담을 예정 00|000000

EventDefine을 먼저 전송하고 정의에따라 아래의 구조체 중 하나를 전송한다.

struct MsgArgu

{

}

struct MonsterSpawnMsg : MsgArgu

{

MonsterType MonsterId;

int MonsterSerialId;

}

struct MonsterHpMsg : MsgArgu

{

MonsterType MonsterId;

int MonsterSerialId;

int Hp;

}

struct PlayerMoveMsg : MsgArgu

{

int PlayerId;

PlayerMove pMove;

}

struct MonsterUseCardMsg : MsgArgu

{

int PlayerId;

CardType card;

}

struct CastleHpMsg : MsgArgu

{

int Hp;

}

게임내에서 발생하는 이벤트를 처리하기위한 이벤트 큐이며 임계영역을 사용하여 관리한다.

std::queue<std::list<BYTE, MsgArgu\*>> EventQueue;

서버에 대기 중인 클라이언트 정보, 두 개라면 방을 생성 후 비운다.

(서버)HANDLE ReadyClient[2];

사용할 함수:

Client

인자 2개를 순서대로 전송

-void Send(BYTE, MsgArgu\*);

패킷을 받아 이벤트 큐로 Push함

-void Recv();

업데이트

이벤트 큐에서 이벤트를 꺼내서 처리

-void EventUpdate();

-void PlayerUpdate();

-void MonsterUpdate();

충돌 확인

void CheckCollision();

업데이트 후 그리기

-void Render();

입력 처리

-void InputCheck();

대기 방 입장 및 시작

void EnterTheRoom();

void StartGame();

BYTE EventDefine을 만들기 위한 함수

-void MakeEventDefine(EventSender sender, EventType type);

Server

인자 2개를 어느 클라이언트에게 보낼 것인가, 안 적으면 둘 다

-void Send(BYTE EventDefine, MsgArgu\* msg, int ClientId = 0);

-void Recv();

플레이어가 다 들어올때까지 기다리기

void WaitPlayer();

몬스터 로직 관련 함수, 여기서 몬스터의 생성 이벤트를 전송할 계획

-void Update();

서버가 accpet후 플레이어 구분을 위한 ID설정

-void SetClientId();

게임이 끝나는 조건인지 체크

-bool IsGameOver();

BYTE EventDefine을 만들기 위한 함수

-void MakeEventDefine(EventSender sender, EventType type);

**팀원 별 역할분담 (구성원 모두가 네트워크 기능 구현에 참여해야 함)**

백종화: 서버 방 스레드 설계, 클라이언트 몬스터 및 카드 관련 설계

김석희: 클라이언트 메인스레드 설계, 클라이언트 대기방 설계

김문수: 서버 메인 스레드 설계, 서버와 클라이언트의 이벤트 처리 스레드 설계

**개발환경**

언어: Python

개발도구: Pycharm Community Edition 2023

운영체제: Windows

**개발일정 (일별/개인별 계획 수립, 달력 형태로 작성)**

백종화: 서버 방 스레드 설계, 클라이언트 몬스터 및 카드 관련 설계

|  |  | 10/31  클라이언트 Card class 설계 | 11/1  클라이언트 Monster class 설계 | 2  클라이언트 Wolf class 설계 | 3 | 4  클라이언트 Bat class 설계 |
| --- | --- | --- | --- | --- | --- | --- |
| 5  1주차 피드백 및 인공지능 시험 준비 | 6  1주차 부족한 진도 보강 | 7  게임수학 시험 준비 | 8  클라이언트 Boss class 설계 (렌더 및 기본적인 생성자) | 9 | 10  클라이언트 Boss class 설계 (공격 패턴) | 11  클라이언트 Boss class 설계 (피격 및 부위파괴, 무력) |
| 12  2주차 피드백 및 부족한 진도 보강 | 13  게임소프트공학 시험 준비 | 14 | 15  서버 방 class 설계  (대기 클라가 2명이었을때 스레드 생성) | 16  서버 방 class 설계  (스레드 생성 후 클라에 번호 부여) | 17  (지스타로 휴식) | 18  (지스타로 휴식) |
| 19  (지스타로 휴식) | 20 | 21  서버 방 class 설계  (이벤트 송수신 처리) | 22  서버 방 class 설계  (몬스터 로직 추가 생성과 그에 따른 이벤트) | 23 | 24  서버 방 게임 초기화 이벤트 관련 로직 구현 | 25  서버 방 게임 시작 이벤트 관련 로직 구현 |
| 26  3, 4주차 피드백 및 부족한 진도 보강 | 27  서버 방 게임 종료 이벤트 관련 로직 구현 | 28 | 29  서버 방 내에 있는 몬스터 식별번호 구현 | 30 | 12/1  식별번호로 메세지 통신 원활한지 확인 | 2 |
| 3  5주차 피드백 및 부족한 진도 보강 | 4  코드 최적화 및 리팩토링 | 5  코드 최적화 및 리팩토링 | 6  디버깅 및 버그수정 | 7  디버깅 및 버그수정 |  |  |

김석희: 클라이언트 메인스레드 설계, 클라이언트 대기방 설계

|  |  | 10/31  대기방 리소스 수집 | 11/1  대기방 씬 구현 | 2  대기방 및 게임 시작관련 이벤트 구현 | 3  Player 관련 리소스 수정 | 4  Input 관련 함수 구현 |
| --- | --- | --- | --- | --- | --- | --- |
| 5  1주차 피드백 | 6  Player class 구현  좌우 이동 등 Input 명령에 대한 구현 | 7  Player class 구현  스킬 사용관련 애니메이션 구현 | 8  Player class 구현  1p, 2p를 구분하기 위한 것들 위주로 구현 | 9  Player class 구현  Player Update 함수 구현 | 10 | 11  Player 관련 이벤트 구현 |
| 12  2주차 피드백 | 13  이벤트 큐 처리 함수 구현 | 14  게임 오버 이벤트 처리 구현 | 15 | 16  Player 이벤트 처리 구현 | 17 | 18 |
| 19 | 20  충돌 체크관련 함수 구현 | 21  충돌 체크 이벤트 구현 | 22  카드와 스킬사용 연결 | 23 | 24  이벤트 관련 코드 정리 | 25 |
| 26  3, 4주차 피드백 | 27  Player와 보스기믹간 상호작용 구현 | 28 | 29  Player의 각성기 관련 구현 | 30 | 12/1  게임 재시작 관련 구현  오브젝트들의 삭제 관련 | 2 |
| 3  5주차 피드백 | 4  게임 재시작 관련 구현 완성 | 5 | 6  디버깅 | 7  디버깅 |  |  |

김문수:

서버 메인 스레드 설계, 서버와 클라이언트의 이벤트 처리 스레드 설계

|  |  | 10/31 | 11/1 | 2  서버 메인 스레드 생성CreatThread() | 3  서버 패킷 수신 구현 | 4  waitPlayer() 구현 |
| --- | --- | --- | --- | --- | --- | --- |
| 5  1주차 피드백 | 6  서버 패킷 송신 구현 | 7  서버 패킷 송신 구현  (eventDefine) | 8  ReadyToPlay() 구현 | 9  event처리 스레드 생성 | 10  클라이언트로부터 수신한  event 자원 관리 설계  Player | 11  클라이언트로부터 수신한  event 자원 관리 설계  Player |
| 12  2주차  비드백 | 13  시험준비 | 14  클라이언트로부터 수신한  event 자원 관리 설계  Monster | 15  시험준비 | 16  시험준비 | 17  코드 정리 및 점검 | 18 |
| 19 | 20  클라이언트로부터 수신한  event 자원 관리 설계  Monster | 21 | 22  event 수신 대기 패킷 구현 | 23  event 수신 대기 패킷 구현 | 24  event 큐 생성 | 25 |
| 26  3, 4주차 피드백 | 27  코드 정리 및 점검 | 28 | 29  클라이언트로부터 들어온 event를 큐에 삽입 구현 | 30 | 12/1  event큐에서 IsGameOver()확인 | 2 |
| 3  5주차  피드백 | 4  event송신 패킷 구현  클라1, 클라2에 송신 | 5  코드 정리 및 점검 | 6  디버깅 및 버그 수정 | 7  디버깅 및 버그 수정 |  |  |