포트폴리오

프로그래머 지원 최예람

CONTENTS

팀 프로젝트 참여

게임 설명

플레이어마다 각 다른 직업을 선택.
플레이어의 직업에 배정된
정령을 이용 전투를 통한
보스전 게임

(3인 멀티 플레이)

PATH TO YIGGDRASIL

PROJECT PYG

플랫폼	PC
장르	액션
카메라	버드뷰
엔진	UNITY

기본 컨셉

보스 섬멸

일반구간 없음
보스전만 진행

역할 수행

탱커, 딜러, 힐러 등
각자의 역할 수행

간접 전투

직접공격 불가
제약이 걸린 간접공격

팀 프로젝트 참여

맡은 업무

C++와 Unity의 서버 구현.

로그인/회원가입 DB 처리.

로그인 – 게임 입장까지의
UI 클라이언트 기능 구현.
서버 기능 구현.

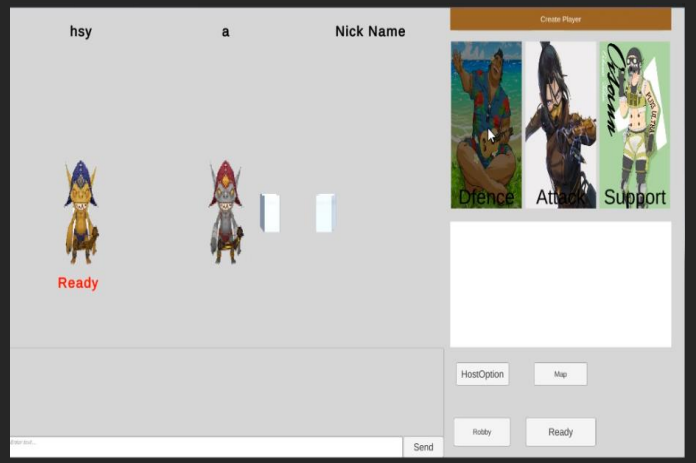
SectorManager 구현.

Excel Reader

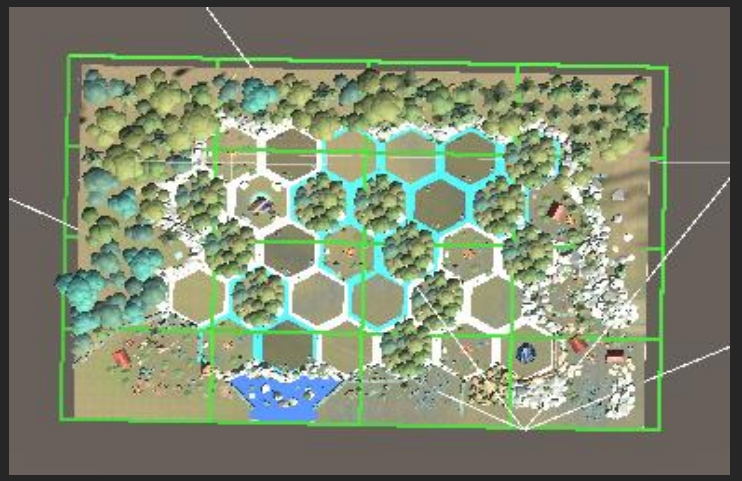
Player 추적 기능 구현

로그인/회원가입

로비 기능
-방 만들기
-방입장
-채팅
-옵션 설정



방 기능
-호스트/멤버 관리
-레디
-캐릭터 선택 (중복 선택 허용 X)
- 맵 선택



게임 기능
-로딩에 필요한 정보
서버-클라 간의 교환 및
데이터 설정.

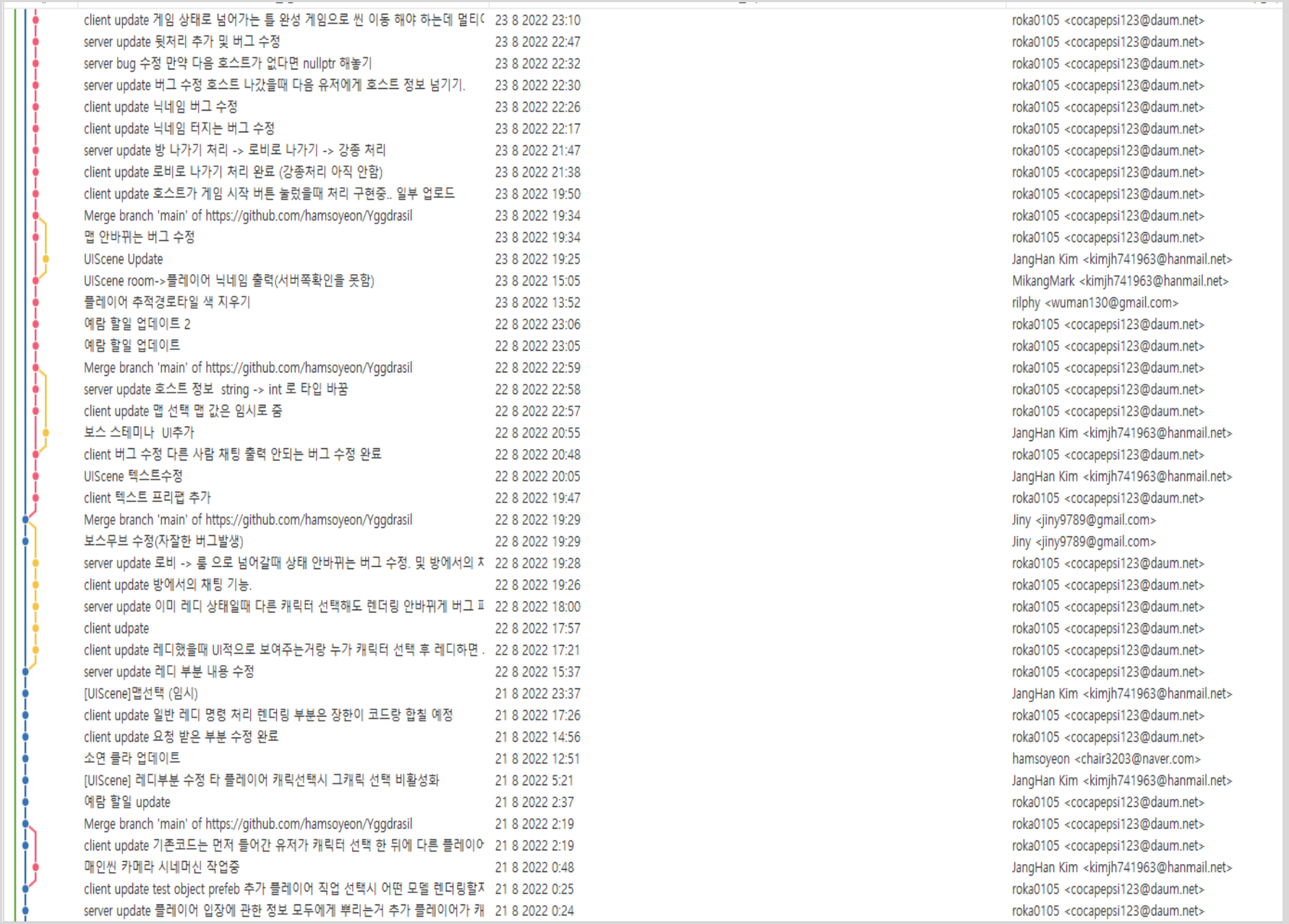
팀 프로젝트 참여

참여도

Roka0105

분배된 역할에 맞게 성실히 임하였으며 팀원들과의 적극적인 소통과 협업을 하였습니다.

필요한 사항, 궁금한 사항이 생길 시 소통을 통해 해결.



client update 게임 상태로 넘어가는 틀 완성 게임으로 씬 이동 해야 하는데 멀티(	23 8 2022 23:10	roka0105 <cocapepsi123@daum.net>
server update 뒷처리 추가 및 버그 수정	23 8 2022 22:47	roka0105 <cocapepsi123@daum.net>
server bug 수정 만약 다음 호스트가 없다면 nullptr 해놓기	23 8 2022 22:32	roka0105 <cocapepsi123@daum.net>
server update 버그 수정 호스트 나갔을때 다음 유저에게 호스트 정보 넘기기.	23 8 2022 22:30	roka0105 <cocapepsi123@daum.net>
client update 닉네임 버그 수정	23 8 2022 22:26	roka0105 <cocapepsi123@daum.net>
client update 닉네임 터지는 버그 수정	23 8 2022 22:17	roka0105 <cocapepsi123@daum.net>
server update 방 나가기 처리 -> 로비로 나가기 -> 강종 처리	23 8 2022 21:47	roka0105 <cocapepsi123@daum.net>
client update 로비로 나가기 처리 완료 (강종처리 아직 안함)	23 8 2022 21:38	roka0105 <cocapepsi123@daum.net>
client update 호스트가 게임 시작 버튼 눌렀을때 처리 구현중.. 일부 업로드	23 8 2022 19:50	roka0105 <cocapepsi123@daum.net>
Merge branch 'main' of https://github.com/hamsoyeon/Yggdrasil	23 8 2022 19:34	roka0105 <cocapepsi123@daum.net>
맵 안바뀌는 버그 수정	23 8 2022 19:34	roka0105 <cocapepsi123@daum.net>
UIScene Update	23 8 2022 19:25	JangHan Kim <kimjh741963@hanmail.net>
UIScene room->플레이어 닉네임 출력(서버쪽확인을 못함)	23 8 2022 15:05	MikangMark <kimjh741963@hanmail.net>
플레이어 추적경로타일 색 지우기	23 8 2022 13:52	rilphy <wuman130@gmail.com>
예람 할일 업데이트 2	22 8 2022 23:06	roka0105 <cocapepsi123@daum.net>
예람 할일 업데이트	22 8 2022 23:05	roka0105 <cocapepsi123@daum.net>
Merge branch 'main' of https://github.com/hamsoyeon/Yggdrasil	22 8 2022 22:59	roka0105 <cocapepsi123@daum.net>
server update 호스트 정보 string -> int 로 타입 바꿈	22 8 2022 22:58	roka0105 <cocapepsi123@daum.net>
client update 맵 선택 맵 값은 임시로 줌	22 8 2022 22:57	roka0105 <cocapepsi123@daum.net>
보스 스테미나 UI추가	22 8 2022 20:55	JangHan Kim <kimjh741963@hanmail.net>
client 버그 수정 다른 사람 채팅 출력 안되는 버그 수정 완료	22 8 2022 20:48	roka0105 <cocapepsi123@daum.net>
UIScene 텍스트수정	22 8 2022 20:05	JangHan Kim <kimjh741963@hanmail.net>
client 텍스트 프리팹 추가	22 8 2022 19:47	roka0105 <cocapepsi123@daum.net>
Merge branch 'main' of https://github.com/hamsoyeon/Yggdrasil	22 8 2022 19:29	Jiny <jiny9789@gmail.com>
보스무브 수정(자잘한 버그발생)	22 8 2022 19:29	Jiny <jiny9789@gmail.com>
server update 로비 -> 룸 으로 넘어갈때 상태 안바뀌는 버그 수정. 및 방에서의 ㅊ	22 8 2022 19:28	roka0105 <cocapepsi123@daum.net>
client update 방에서의 채팅 기능.	22 8 2022 19:26	roka0105 <cocapepsi123@daum.net>
server update 이미 레디 상태일때 다른 캐릭터 선택해도 렌더링 안바뀌게 버그 ㅍ	22 8 2022 18:00	roka0105 <cocapepsi123@daum.net>
client udpate	22 8 2022 17:57	roka0105 <cocapepsi123@daum.net>
client update 레디했을때 UI적으로 보여주는거랑 누가 캐릭터 선택 후 레디하면 .	22 8 2022 17:21	roka0105 <cocapepsi123@daum.net>
server update 레디 부분 내용 수정	22 8 2022 15:37	roka0105 <cocapepsi123@daum.net>
[UIScene]맵선택 (임시)	21 8 2022 23:37	JangHan Kim <kimjh741963@hanmail.net>
client update 일반 레디 명령 처리 렌더링 부분은 장한이 코드랑 합칠 예정	21 8 2022 17:26	roka0105 <cocapepsi123@daum.net>
client update 요청 받은 부분 수정 완료	21 8 2022 14:56	roka0105 <cocapepsi123@daum.net>
소연 클라 업데이트	21 8 2022 12:51	hamsoyeon <chair3203@naver.com>
[UIScene] 레디부분 수정 타 플레이어 캐릭선택시 그캐릭 선택 비활성화	21 8 2022 5:21	JangHan Kim <kimjh741963@hanmail.net>
예람 할일 update	21 8 2022 2:37	roka0105 <cocapepsi123@daum.net>
Merge branch 'main' of https://github.com/hamsoyeon/Yggdrasil	21 8 2022 2:19	roka0105 <cocapepsi123@daum.net>
client update 기존코드는 먼저 들어간 유저가 캐릭터 선택 한 뒤에 다른 플레이어	21 8 2022 2:19	roka0105 <cocapepsi123@daum.net>
매인씬 카메라 시네머신 작업중	21 8 2022 0:48	JangHan Kim <kimjh741963@hanmail.net>
client update test object prefeb 추가 플레이어 직업 선택시 어떤 모델 렌더링할지	21 8 2022 0:25	roka0105 <cocapepsi123@daum.net>
server update 플레이어 입장에 관한 정보 모두에게 뿌리는거 추가 플레이어가 캐	21 8 2022 0:24	roka0105 <cocapepsi123@daum.net>

Render Sector

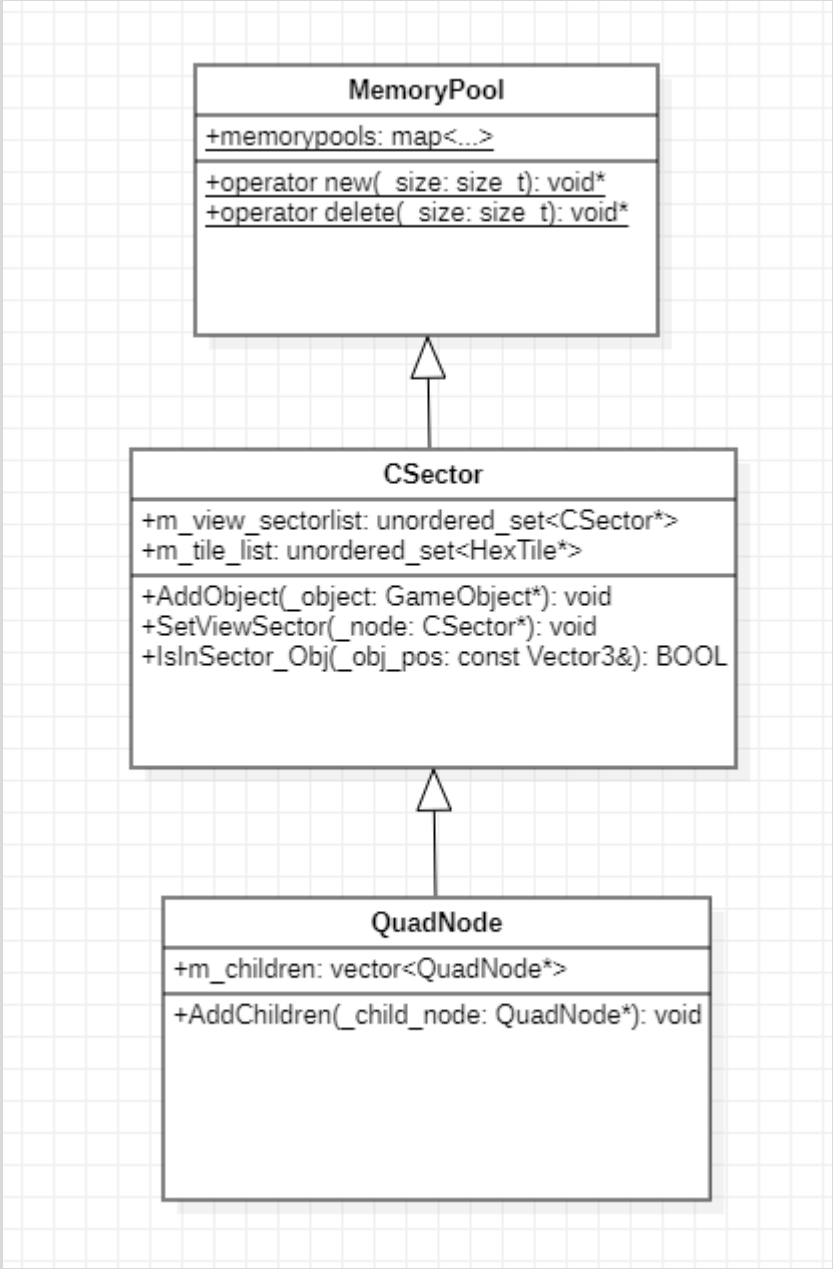
Sector 구조

QuadNode 는
Children 정보를 관리

Sector 는
현재 Sector의 시야 내에
들어오는 타일들의 정보와
Sector들의 정보를
관리 합니다.

+ 오브젝트들의 정보는
Sector를 통하여
해당하는 타일에
등록됩니다.

확장성을 위해 추후에
Csector와 QuadNode의 상속구조를
바꿀 예정입니다.




```
#include "MemoryPool_2.h"
class GameObject;
class HexTile;
class CSector :public MemoryPool_2
{
protected:
    CSector();
    CSector(Vector3 _senter_pos,Vector3 _distance);

    virtual ~CSector();
public:
    void AddObject(GameObject* _object);      // 오브젝트 등록
    const Vector3 GetDistance();
    void SetDistance(Vector3 _pos);
    const Vector3 GetSenter();
    const Vector3 GetStartPos();
    BOOL IsInSector_Direction(const Vector3 _obj_pos,E_NodeType _type);
    BOOL IsInSector(const Vector3 _obj_pos);// 오브젝트가 노드안에 있는지 체크
    BOOL IsInSector_Obj(const Vector3& _obj_pos);
    void SetViewSector(CSector* _node);
    unordered_set<CSector*>& GetViewSector();
    unordered_set<HexTile*>& GetTileList();
private:
    unordered_set<CSector*> m_view_sectorlist;
    unordered_set<HexTile*> m_tile_list;
    Vector3 m_start_pos;
    Vector3 m_senter_pos;         // 노드의 중심 위치
    Vector3 m_distance;           // 밑변/2
};
class QuadNode : public CSector
{
public:
    static int GetCreateCount();
public:
    QuadNode();
    QuadNode(Vector3 _senter_pos, Vector3 _distance);
    virtual ~QuadNode()final;
    void AddChildren(QuadNode* _child_node);// 자식노드 등록
    QuadNode* GetChildNode(int index);              // 자식노드 가져오기

    void SetID();
    int GetID();
    QuadNode* GetParent();                // 부모노드 가져오기

    int Child_Size();
private:
    static int create_count;
    int m_id;
    vector<QuadNode*> m_children; // 자식노드
    QuadNode* m_parent;           // 부모노드
    BOOL m_is_culled;             // 컬링 여부
    const float m_limit_depth = 4;// 트리의 최대 깊이
    E_NodeType m_type;
};
```

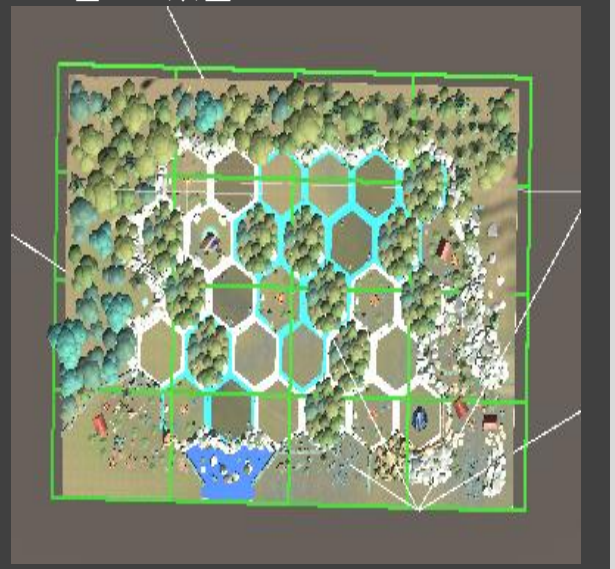
사용 언어 : C++

Render Sector

SectorManager

Sector Manager 에서
Game에 돌입했을 때
해당 게임의 맵에 대한
Sector와 그 자식들의 값을
설정합니다.

서버에서 충돌 처리를 하게 될
경우를 생각해 Quad Tree로
구현 하였습니다.



초록선=sector / 현재 depth:2

CSectorMgr
+m_roots: map<...>
-CreateQuadTree(...): void -CreateChildren(_root: QuadNode*, ...): void -SerchObjectNode(_root: QuadNode*, ...): QuadNode** +AddQuadTree(...): void +SerchObjectNode(_gameinfo: t_GameInfo*, _pos: Vector3): QuadNode* +SendViewSectorFunc(...): void +SendViewTileFunc(...): void +PlayerMoveProcess(...): void

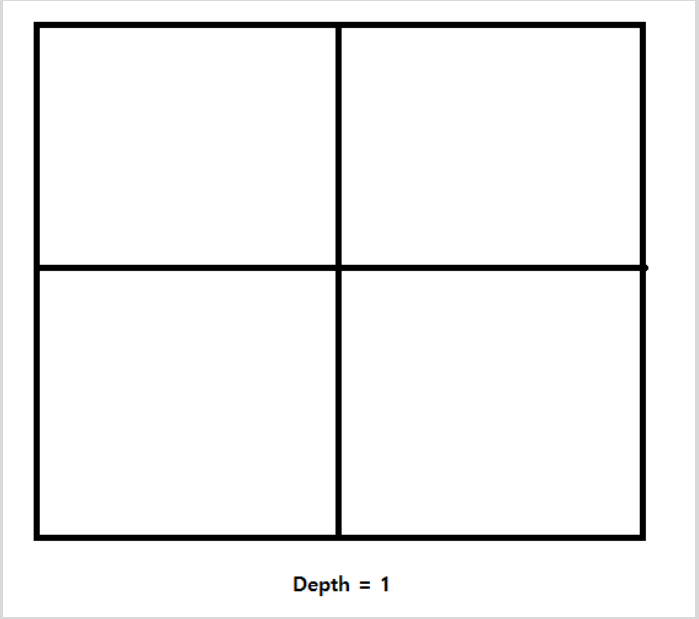
```
#pragma once
#include "pch.h"
#include "CSector.h"
#include "CGameMgr.h"

class CSector;
class GameObject;
class CSession;
class CSectorMgr
{
public:
    static CSectorMgr* GetInst();
    static void Create();
    static void Destory();
private:
    static CSectorMgr* m_instance;
    CSectorMgr();
    ~CSectorMgr();

    void CreateQuadTree(t_GameInfo* _gameinfo, t_MapInfo* _mapinfo);
    void CreateChildren(QuadNode* _parent, Vector3 _senterpos, Vector3 _distance, int _curdepth, t_GameInfo* _gameinfo);

    QuadNode** SerchNode(QuadNode* _parent, int _id, int _curdepth, t_MapInfo* _mapinfo);
    void SetViewNode(QuadNode* _parent, int _curdepth, t_MapInfo* _mapinfo, UINT _gameid);
    QuadNode** SerchNode(QuadNode* _parent, Vector3 _pos, int _curdepth, t_MapInfo* _mapinfo);
    QuadNode** SerchObjectNode(QuadNode* _parent, Vector3 _pos, int _curdepth, t_MapInfo* _mapinfo);

    void SendViewSectorFunc(CSession* _session, t_GameInfo* _gameinfo, Vector3 _objpos);
    void SendViewTileFunc(CSession* _session, t_GameInfo* _gameinfo, Vector3 _objpos);
public:
    void Init();
    void End();
    void AddQuadTree(t_GameInfo* _gameinfo,t_MapInfo* _mapinfo);
    void SendInit(UINT _gameid, CSession* _session, t_MapInfo* _mapinfo);
    QuadNode* SerchObjectNode(t_GameInfo* _gameinfo,Vector3 _pos);
    //test
    QuadNode* return_root(UINT gameid)
    {
        return m_roots[gameid];
    }
    /* ... */

    void PlayerMoveProcess(CSession* _session, t_GameInfo* _gameinfo);
public:
    void Packing(unsigned long _protocol, Vector3 _startpos, Vector3 _endpos, float _h_distance, float _v_distance, int _sectorcount, CSession* _session);
    void Packing(unsigned long _protocol, list<Vector3>& _starts, Vector3& _distances, CSession* _session);
    void UnPacking(byte* _recvbuf, Vector3& _pos);
private:
    CLock* m_lock;
    RBT<UINT, QuadNode*> m_roots; // key = gameinfo id,  value = rootnode

};
```

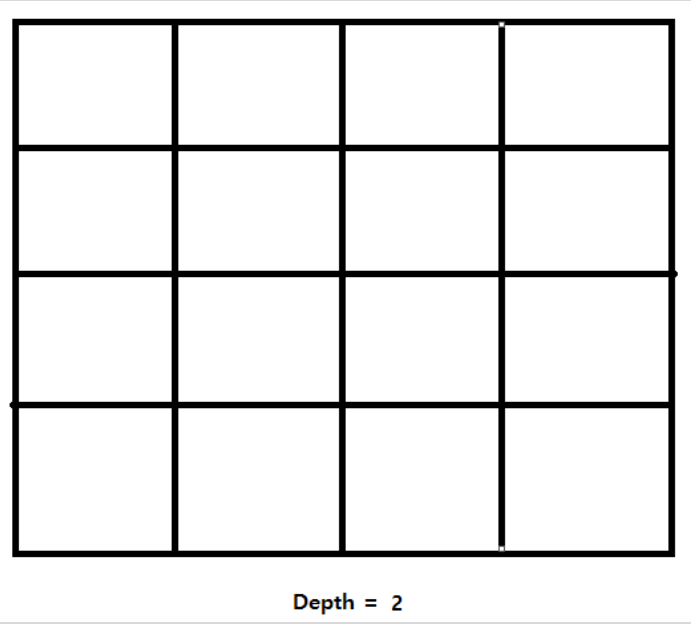
사용 언어 : C++

Render Sector

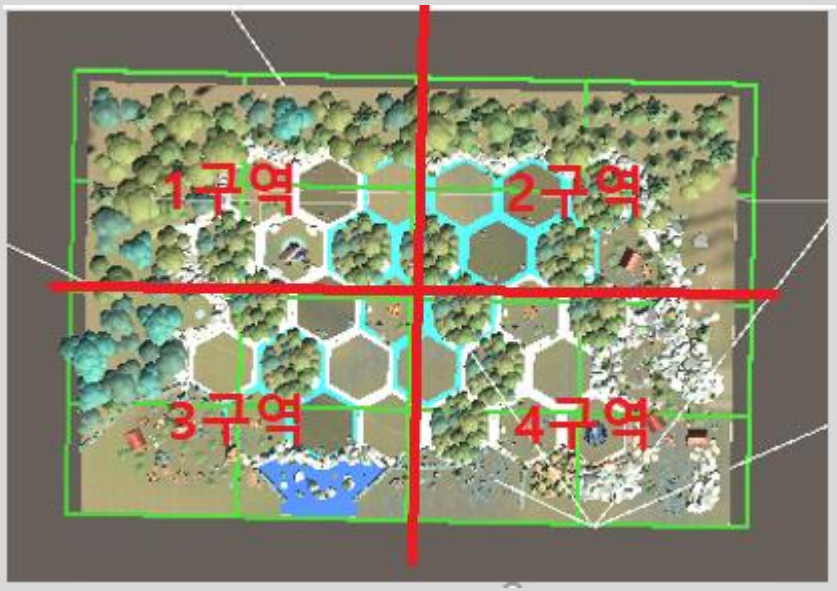
Quad Tree

Depth값을 주어 해당 값 수정 시 트리가 4의 depth승 만큼 구역이 쪼개집니다.





유효한 데이터는 제일 깊이 있는 노드들 이며, 상위 노드들은 구역의 의미만을 갖습니다.

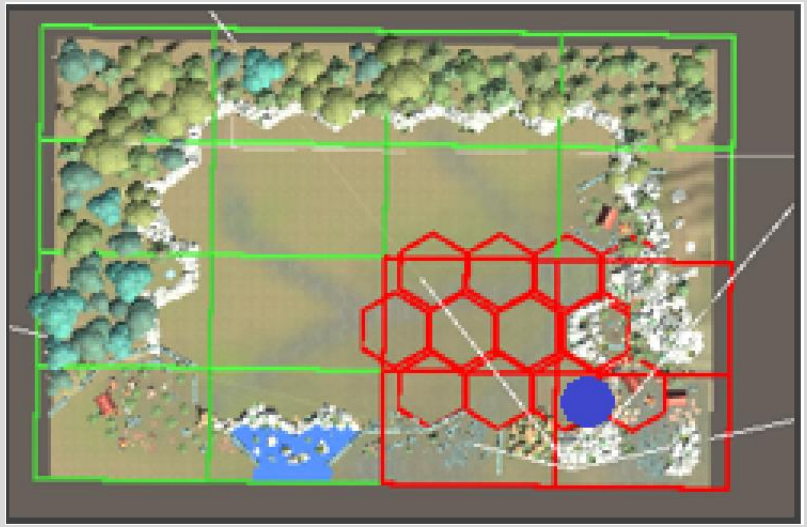


Render Sector

Quad Tree

Player의 위치를 받아와
해당 위치에 속하는
Sector를 탐색.

해당 Sector와 Tile 정보를
잘 읽어서 클라에 전송했는지
확인을 위해 해당 범위를
빨갛게 변하도록 처리했습니다.



Player 위치 = 파란 원

```
QuadNode* CSectorMgr::SerchObjectNode(t_GameInfo* _gameinfo, Vector3 _pos)
{
    QuadNode* node = reinterpret_cast<QuadNode*>(*SerchObjectNode(m_roots[_gameinfo->m_id], _pos, 0, _gameinfo->m_mapinfo));
    return node;
}
#pragma endregion
```

```
QuadNode** CSectorMgr::SerchObjectNode(QuadNode* _parent, Vector3 _pos, int _curdepth, t_MapInfo* _mapinfo)
{
    if (_mapinfo->m_depth == _curdepth)
    {
        if (_mapinfo->m_start_position.x <= _pos.x && _mapinfo->m_end_position.x >= _pos.x
            && _mapinfo->m_start_position.z >= _pos.z && _mapinfo->m_end_position.z <= _pos.z)

        {
            //if (_parent->IsInSector_Direction(_pos, _type))
            //{

            return &_parent;
            //}
        }
        return nullptr;
    }
    QuadNode* child = nullptr;
    QuadNode** item = nullptr;
    int size = _parent->Child_Size();
    for (int i = 0; i < size; i++)
    {
        child = _parent->GetChildNode(i);
        if (child == nullptr)
        {
            return nullptr;
        }
        if (child->IsInSector_Obj(_pos))
            item = SerchObjectNode(child, _pos, _curdepth + 1, _mapinfo);
        if (item != nullptr)
            return item;
    }
    return nullptr;
}
```

위치값 기반으로 Sector 검색

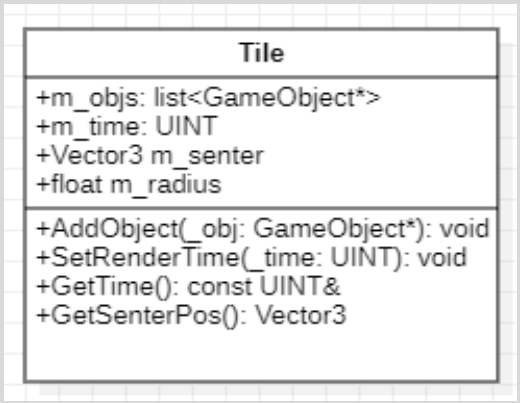
사용 언어 : C++

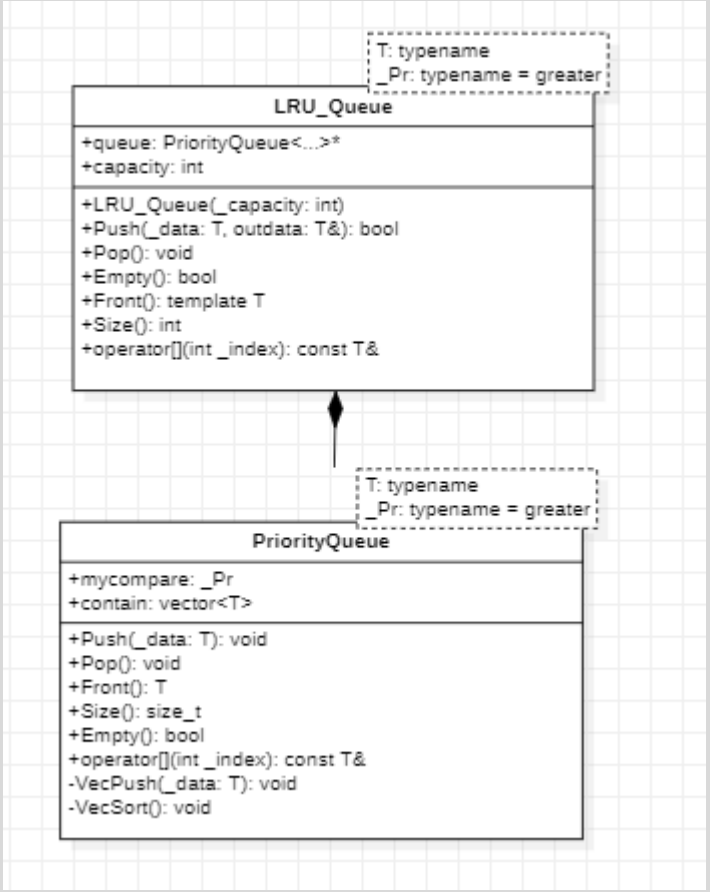
Render Sector

LRU Queue

우선순위 큐를 구현하여
렌더링 범위에 들어올 때 마다
해당 Tile에 대한 time 변수를
그때의 시간으로 갱신해 줍니다.

불 필요하게 렌더링을 ON/OFF
비용을 줄이기 위해
Queue의 용량 만큼은
렌더링을 유지하되
새로운 오브젝트가
추가로 들어왔을 때는
가장 시간 갱신이 안된
오래된 오브젝트부터 렌더링에서
제외시킵니다.






```cpp
void VecPush(T _data)
{
    contain.push_back(_data);

    for (int i = 1; i < contain.size(); i++)
    {
        int child = i;
        do
        {
            int root = (child - 1) / 2;
            if (mycompare(contain[root], contain[child]))
            {
                T temp = contain[root];
                contain[root] = contain[child];
                contain[child] = temp;
            }
            child = root;
        } while (child != 0);
    }
}
void VecSort()
{
    if (contain.size() == 1)
    {
        return;
    }
    for (int i = contain.size() - 1; i >= 0; i--)
    {
        T temp = contain[0];
        contain[0] = contain[i];
        contain[i] = temp;
        int root = 0;
        int child = 1;
        do
        {
            child = root * 2 + 1;
            if (this->mycompare(contain[child], contain[child + 1]) && child < i - 1)
            {
                child++;
            }
            if (this->mycompare(contain[root], contain[child]) && child < i)
            {
                temp = contain[root];
                contain[root] = contain[child];
                contain[child] = temp;
            }
            root = child;
        } while (child < i);
    }
}
```

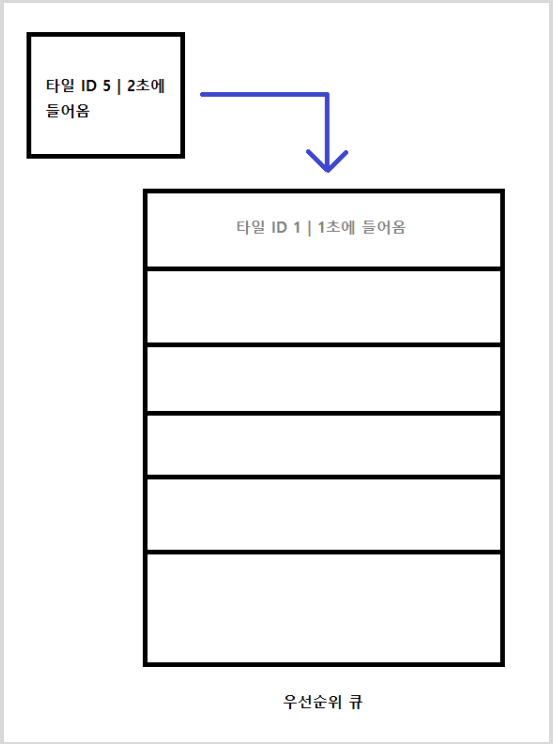
Priority Queue.h

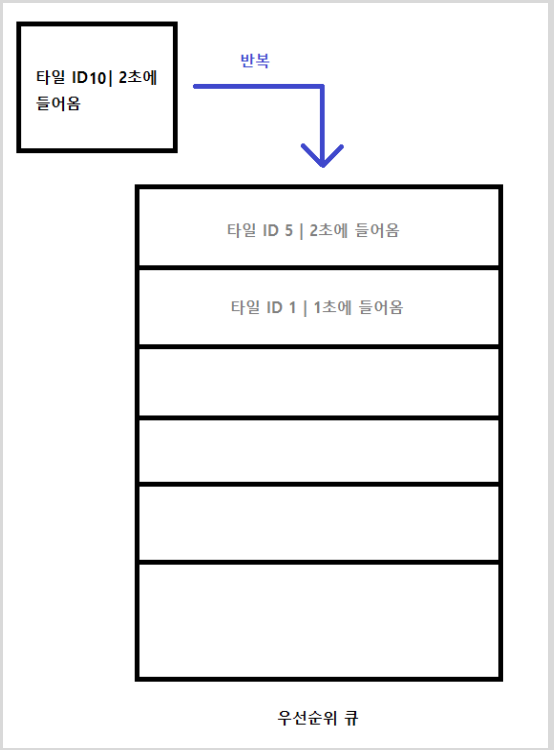
사용 언어 : C++

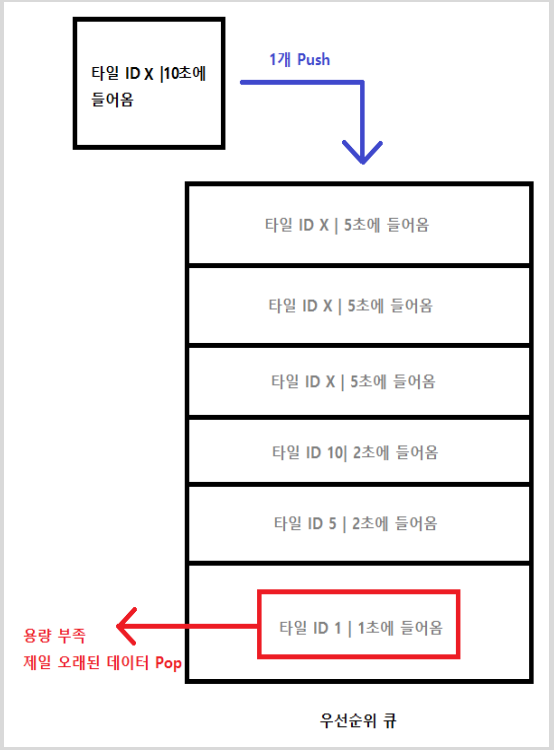
Render Sector

LRU Queue

갱신이 오랫동안 안되어 있는데
Queue가 꽉 찬 경우
갱신 시점이 오래된 Tile 부터
Queue 에서 제외 시켜 줍니다.







Render Sector

LRU Queue

각 Session 들이
렌더중인 Sector 정보가
다를 것 이기 때문에
Session 마다 LRU Queue를
소유하게 했습니다.

실행 화면(Server)은
제외된 타일의 ID와
해당 타일이 렌더링
시작된시점의 시간을 출력합니다.

실행 화면(Client)은
검정 원->빨간 원의 위치로
이동했을 때 시야 범위이며

겹치는 부분의 오브젝트=시간갱신!
아닌 부분=제외될 수 있음!

CSession.h

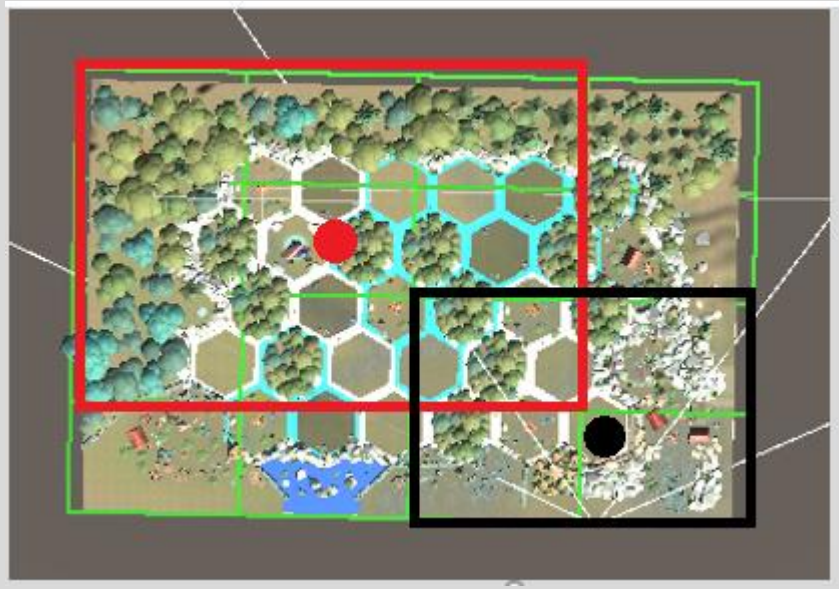
```
LRU_Queue<HexTile*, mygreater>* m_real_queue;
LRU_Queue<HexTile*,mygreater>* m_temp_queue;

void SetSector(QuadNode* _nodesector)
{
    m_sector = _nodesector;
    double curtime = 0;
    time_t last_update_time = time(NULL);
    unordered_set<CSector*>& viewlist = m_sector->GetViewSector();

    bool same = false;
    unordered_set<HexTile*> tilelist;
    list<HexTile*> suc_tile;
    list<HexTile*> new_tile;
    while (m_real_queue->Empty() == false)
    {
        same = false;
        HexTile* temp = m_real_queue->Front();
        m_real_queue->Pop();
        for (auto sector : viewlist) { ... }
        if (!same)
        {   //리얼 큐에 있고 타일 리스트에 없는 경우=원래 시간으로 넣기.
            //다 돌았는데 tilelist에 없었다. real queue에만 있었다.
            HexTile* data = nullptr;
            if (m_temp_queue->Push(temp, data))
            {
                cout << data->GetTime() << " 이거 지워짐\n";
                delete data;
            }
        }
    }

    //리얼 큐에 없고 타일 리스트에만 있는경우
    for (auto sector : viewlist) { ... }

    LRU_Queue<HexTile*,mygreater>* tempqueue;
    tempqueue = m_real_queue;
    m_real_queue = m_temp_queue;
    m_temp_queue = tempqueue;
}
```

Queue의 렌더링 정보를 갱신하는 코드 입니다.

TileID | 갱신 시간

```
36|1663167027 이거 지워짐
37|1663167027 이거 지워짐
22|1663167027 이거 지워짐
23|1663167027 이거 지워짐
31|1663167027 이거 지워짐
38|1663167027 이거 지워짐
39|1663167027 이거 지워짐
12|1663167062 이거 지워짐
13|1663167062 이거 지워짐
14|1663167062 이거 지워짐
1|1663167062 이거 지워짐
2|1663167062 이거 지워짐
3|1663167062 이거 지워짐
```

실행 화면 (Server)



실행 화면(Client)

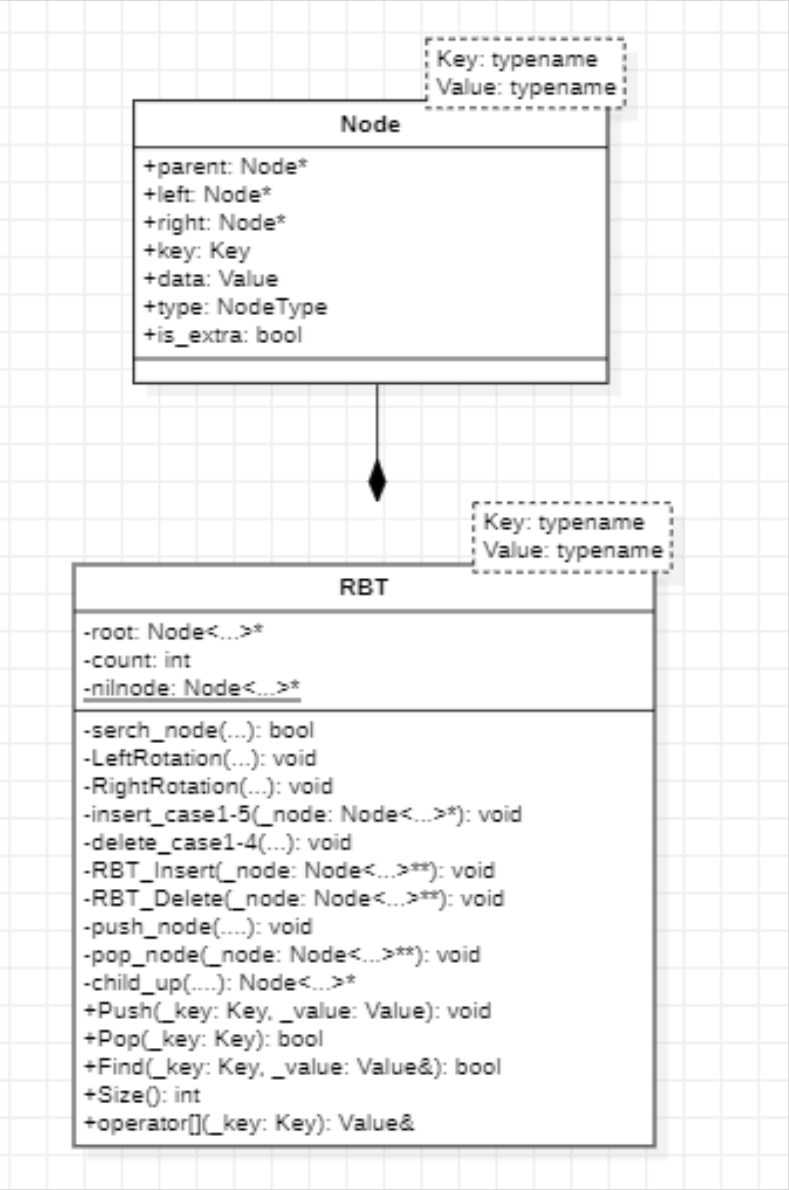
사용 언어 : C++

Map

Red-Black-Tree

Red-Black-Tree 알고리즘을
학습하여 완전 이진 트리를
항상 유지하도록 하여
검색 시간을 단축하도록 했습니다

하지만 템플릿 특수화는
적용하지 않았기 때문에
key값은 int 형만 가능합니다.




```cpp
enum class NodeType
{
    None,
    Black,
    Red
};
```

노드 멤버 변수 Type Enum

```cpp
template<typename Key, typename Value>
class Node
{
```

```cpp
template <typename Key, typename Value>
class Node<Key, Value*>
```

노드에 대한 특수화
(포인터 데이터일 경우)

```cpp
public:
    RBT() { ... }
    void Push(Key _key, Value  _value)
    {
        Node<Key, Value>* node = new Node<Key, Value>(_key, _value, nilnode, NodeType::Red);
        push_node(&root, node);
        RBT_Insert(&node);
        count++;
    }
    bool Pop(Key _key)
    {
        Node<Key, Value>** serchnode;
        if (serch_node(&root, _key, serchnode))
        {
            pop_node(serchnode);
            count--;
            return true;
        }
        return false;
    }
    bool Find(Key _key, Value& _value) { ... }
    int Size() { ... }
    Value& { ... }
};
```

RBT.h public 함수

사용 언어 : C++

Map

Red- Black- Tree

구현한 STL을 map으로 사용.

```
ublic:
   static RBT<DWORD, t_ThreadInfo*> g_threadinfo;
rotected:
```
Clocp.h* ✕

Clocp.h
= 스레드ID,스레드 정보

```
RBT<UINT,t_MapInfo*> m_maps;
```
CMapMgr.h ✕

CMapMgr.h
= 맵ID,맵 정보

```
RBT<UINT, QuadNode*> m_roots; // key = gameinfo id,  value = rootnode
```
CSectorMgr.h ✕

CSectorMgr.h
= GameID,root Node주소

```
static RBT</*객체크기*/UINT,/*pool 정보*/t_pool_info*> memorypools;
```
MemoryPool_2.h ✕

MemoryPool_2.h
= 메모리 크기,생성된 memorypool 정보

사용 언어 : C++

MemoryPool

동작 방식

메모리 조각화 없이 재사용하면서
사용하기 위해서 구현했습니다.

1.Operator new /delete
사용으로 메모리 풀을 구현.

2.사용 요청이 들어온 데이터를
일정 단위로 구분된 메모리풀 중
해당되는 단위의
메모리풀을 배정.
(생성이 안 되어있다면 최초
사용할 때 생성해서 map에 등록)
배정된 size = map key

3.그 메모리풀에서
메모리 블록을 할당 받는다.

```
static RBT</*객체크기*/UINT,/*pool 정보*/t_pool_info*> memorypools;
```

```
#define MAXMEMORY_BYTE 32768

#include <stdio.h>
#include <stdlib.h>
#include <Windows.h>
#include <vector>
#include <map>

using namespace std;

struct t_pool_info
{
    t_pool_info(const int _capacity, const int _size) { ... }
    ~t_pool_info() { ... }
    char* current;
    //이미 생성된 블럭이 꽉 차서 새로운 블럭을 만들었을 때 그 블럭의 시작 주소.
    vector<char*> startptr;

};


class MemoryPool_2
{
public:
    static void* operator new(size_t _size);
    static void* operator new[](size_t size);

    static void operator delete(void* _object, size_t _size);

    static int AssignSize(size_t _size);
    static void End();

protected:
    static RBT</*객체크기*/UINT,/*pool 정보*/t_pool_info*> memorypools;

};
```

Memorypool_2.h

사용 언어 : C++

MemoryPool

AssignSize()

만약 1byte의 메모리 요청이 들어온다면 8byte memorypool 에서 메모리 블록을 반환 합니다.

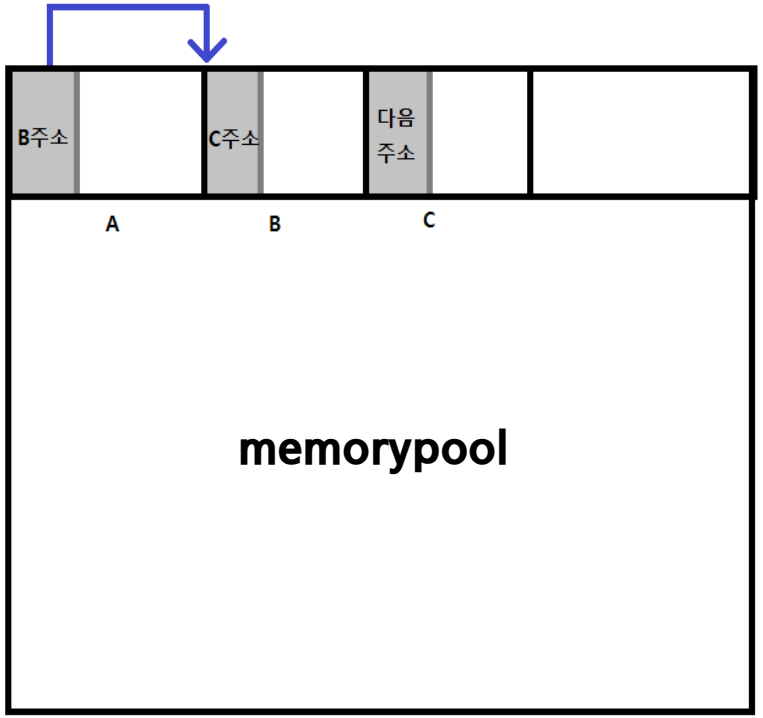
8byte부터
2의 제곱만큼 size를 할당하는 이유
64bit 기준 포인터 크기 = 8byte

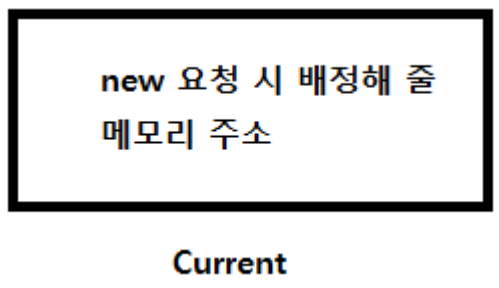
현재 메모리풀은 각 블록에서 다음 할당 시 호출할 메모리 주소가 쓰여져 있습니다.

따라서 메모리 블록의 최소 크기는 8byte 여야 했습니다.

또 1byte 단위로 다르다고 모두 Memorypool을 만들어주는것은 해당 단위의 객체가 많이 사용 되지 않을 경우 낭비가 되기 때문에 일정 단위로 pool을 만들어 사용하도록 하였습니다.

```
int MemoryPool_2::AssignSize(size_t _size)
{
    if (_size * 2 > MAXMEMORY_BYTE)
        return 0;
    int befor = 0;
    for (int i = 8; i <= MAXMEMORY_BYTE / 2; i *=2)
    {
        if (_size > befor && _size <= i)
        {
            return i;
        }
        befor = i;
    }

    //음수인 경우
    return -1;
}
```





현재는 A 주소가 들어감.
A 를 할당 해 주고
A의 8byte를 읽어와
다음 주소를 다시
current에 넣어줍니다.

사용 언어 : C++

MemoryPool

New/Delete

NEW

모든 블록은
MEMORY_BYTE=32768
만큼 할당 받고 해당 블록에서
쪼개서 사용합니다.

최초 할당 or 사용중인 블록이 용량X
=
새로운 블록 추가 할당

DELETE

반환된 memory를
memorypool에
반환합니다.

New

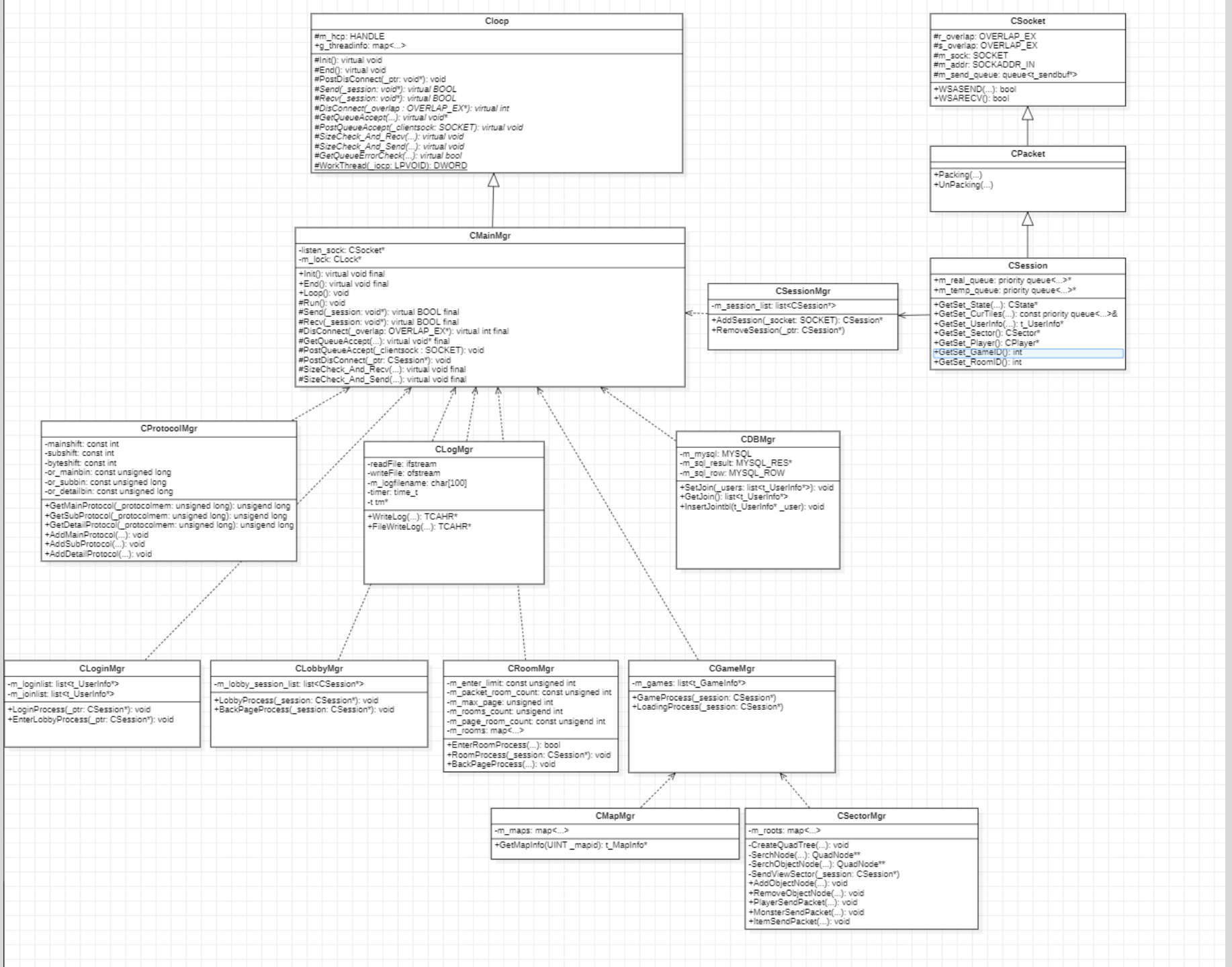
```
void* MemoryPool_2::operator new(size_t _size)
{
    char* next_ptr = nullptr;
    int size = AssignSize(_size);
    if (size == -1)
    {
        //뭔가 잘못 됨.
        return nullptr;
    }
    else if (size == 0)
    {
        // MAXMEMORY_BYTE/2 보다 값이 큰 경우 메모리풀을 생성하는게 낭비이기 때문에
        // 그 byte size 자체로 new 를 해준다.
        return malloc(_size);
    }
    t_pool_info* dummy=nullptr;
    if (memorypools.Find(size,dummy)==false)
    {
        //키 못찾음
        memorypools.Push(size, new t_pool_info(MAXMEMORY_BYTE, size));
    }
    else
    {
        //키 찾았는데 current 가 null 인 경우(이미 모두 할당한 경우 새로운 블럭 추가로 받기)
        if (memorypools[size]->current == nullptr)
        {
            char** ptr;
            char* ptr2;
            memorypools[size]->current = (char*)malloc(MAXMEMORY_BYTE);
            memset(memorypools[size]->current, 0, MAXMEMORY_BYTE);

            next_ptr = (char*)memorypools[size]->current;
            memorypools[size]->startptr.push_back(memorypools[size]->current);

            for (int i = 1; i < MAXMEMORY_BYTE / size; i++)
            {
                ptr2 = memorypools[size]->current + (size * i);
                ptr = &ptr2;
                memcpy(next_ptr, ptr, sizeof(char*));
                next_ptr += _size;
            }
            ptr2 = nullptr;
            ptr = &ptr2;
            // 맨 마지막에 nullptr 넣기
            memcpy(next_ptr, ptr, sizeof(char*));
        }

    }
    char* curptr = memorypools[size]->current;
    char* result = curptr;


    memcpy(&next_ptr, curptr, sizeof(char*));
    memorypools[size]->current = next_ptr;

    return result;
}
```

Delete

```
void MemoryPool_2::operator delete(void* _object, size_t _size)
{
    int size = AssignSize(_size);
    if (size == -1)
    {
        //뭔가 잘못 됨.
        return;
    }
    else if (size == 0)
    {
        // MAXMEMORY_BYTE/2 보다 값이 큰 경우 메모리풀을 생성하는게 낭비이기 때문에
        // 그 byte size 자체로 new 를 했기 때문에 그 메모리 자체를 delete 해준다.
        free(_object);
        return;
    }
    char* curptr = memorypools[size]->current;
    memcpy((char*)_object, &curptr, sizeof(char*));
    memorypools[size]->current = (char*)_object;
}
```

사용 언어 : C++

IOCP 전체 구조


Ciocp
#m_hcp: HANDLE
+g_threadInfo: map<...>
#Init(): virtual void
#End(): virtual void
#PostDisConnect(_ptr: void*): void
#Send(_session: void*): virtual BOOL
#Recv(_session: void*): virtual BOOL
#DisConnect(_overlap : OVERLAP_EX*): virtual int
#GetQueueAccept(...): virtual void*
#PostQueueAccept(_clientsock: SOCKET): virtual void
#SizeCheck_And_Recv(...): virtual void
#SizeCheck_And_Send(...): virtual void
#GetQueueErrorCheck(...): virtual bool
#WorkThread(_iocp: LPVOID): DWORD
CSocket
#r_overlap: OVERLAP_EX
#s_overlap: OVERLAP_EX
#m_sock: SOCKET
#m_addr: SOCKADDR_IN
#m_send_queue: queue<t_sendbuf*>
+WSASEND(...): bool
+WSARECV(): bool
CPacket
+Packing(...)
+UnPacking(...)
CMainMgr
-listen_sock: CSocket*
-m_lock: CLock*
+Init(): virtual void final
+End(): virtual void final
+Loop(): void
#Run(): void
#Send(_session: void*): virtual BOOL final
#Recv(_session: void*): virtual BOOL final
#DisConnect(_overlap: OVERLAP_EX*): virtual int final
#GetQueueAccept(...): virtual void* final
#PostQueueAccept(_clientsock : SOCKET): void
#PostDisConnect(_ptr: CSession*): void
#SizeCheck_And_Recv(...): virtual void final
#SizeCheck_And_Send(...): virtual void final
CSessionMgr
-m_session_list: list<CSession*>
+AddSession(_socket: SOCKET): CSession*
+RemoveSession(_ptr: CSession*)
CSession
+m_real_queue: priority queue<...>*
+m_temp_queue: priority queue<...>*
+GetSet_State(...): CState*
+GetSet_CurTiles(...): const priority queue<...>&
+GetSet_UserInfo(...): t_UserInfo*
+GetSet_Sector(): CSector*
+GetSet_Player(): CPlayer*
+GetSet_GameID(): int
+GetSet_RoomID(): int
CProtocolMgr
-mainshift: const int
-subshift: const int
-byteshift: const int
-or_mainbin: const unsigned long
-or_subbin: const unsigned long
-or_detailbin: const unsigned long
+GetMainProtocol(_protocolmem: unsigned long): unsigend long
+GetSubProtocol(_protocolmem: unsigned long): unsigend long
+GetDetailProtocol(_protocolmem: unsigned long): unsigend long
+AddMainProtocol(...): void
+AddSubProtocol(...): void
+AddDetailProtocol(...): void
CLogMgr
-readFile: ifstream
-writeFile: ofstream
-m_logfilename: char[100]
-timer: time_t
-t tm*
+WriteLog(...): TCAHR*
+FileWriteLog(...): TCAHR*
CDBMgr
-m_mysql: MYSQL
-m_sql_result: MYSQL_RES*
-m_sql_row: MYSQL_ROW
+SetJoin(_users: list<t_UserInfo*>): void
+GetJoin(): list<t_UserInfo*>
+InsertJointol(t_UserInfo* _user): void
CLoginMgr
-m_loginlist: list<t_UserInfo*>
-m_joinlist: list<t_UserInfo*>
+LoginProcess(_ptr: CSession*): void
+EnterLobbyProcess(_ptr: CSession*): void
CLobbyMgr
-m_lobby_session_list: list<CSession*>
+LobbyProcess(_session: CSession*): void
+BackPageProcess(_session: CSession*): void
CRoomMgr
-m_enter_limit: const unsigned int
-m_packet_room_count: const unsigned int
-m_max_page: unsigned int
-m_rooms_count: unsigend int
-m_page_room_count: const unsigend int
-m_rooms: map<...>
+EnterRoomProcess(...): bool
+RoomProcess(_session: CSession*): void
+BackPageProcess(...): void
CGameMgr
-m_games: list<t_GameInfo*>
+GameProcess(_session: CSession*)
+LoadingProcess(_session: CSession*)
CMapMgr
-m_maps: map<...>
+GetMapInfo(UINT _mapid): t_MapInfo*
CSectorMgr
-m_roots: map<...>
-CreateQuadTree(...): void
-SerchNode(...): QuadNode**
-SerchObjectNode(...): QuadNode**
-SendViewSector(_session: CSession*)
+AddObjectNode(...): void
+RemoveObjectNode(...): void
+PlayerSendPacket(...): void
+MonsterSendPacket(...): void
+ItemSendPacket(...): void

IOCP 비동기화

IOCP를 사용하여
소켓 함수들을
비동기화

```
DWORD CIocp::WorkThread(LPVOID _iocp)
{
    int retval;

    CIocp* ciocp = reinterpret_cast<CIocp*>(_iocp);
    t_ThreadInfo* myinfo = new t_ThreadInfo();
    g_threadinfo.Push(GetCurrentThreadId(),myinfo);

    while (1)
    {
        DWORD cbTransferred;
        SOCKET clientsock;
        OVERLAP_EX* overlap_ptr;
        void* session;
        retval = GetQueuedCompletionStatus(ciocp->m_hcp, &cbTransferred, &clientsock, (LPOVERLAPPED*)&overlap_ptr, INFINITE);

        bool check = ciocp->GetQueueErrorCheck(retval, cbTransferred, overlap_ptr);
        // check에서 에러체크후 오류발생시 type을 disconnected로 바꾸어준다.
        switch (overlap_ptr->type)
        {
        case IO_TYPE::ACCEPT:
            session = ciocp->GetQueueAccept(clientsock, overlap_ptr);
            ciocp->Recv(session);
            break;
        case IO_TYPE::RECV:
            session = overlap_ptr->session;
            ciocp->SizeCheck_And_Recv(session, cbTransferred,myinfo); // 여기서 함수하나에 전부 처리하도록
            break;
        case IO_TYPE::SEND:
            session = overlap_ptr->session;
            ciocp->SizeCheck_And_Send(session, cbTransferred,myinfo);
            break;
        case IO_TYPE::DISCONNECT:
            ciocp->DisConnect(overlap_ptr);
            break;
        }
    }

    return 0;
}
```

사용 언어 : C++

IOCP State 패턴

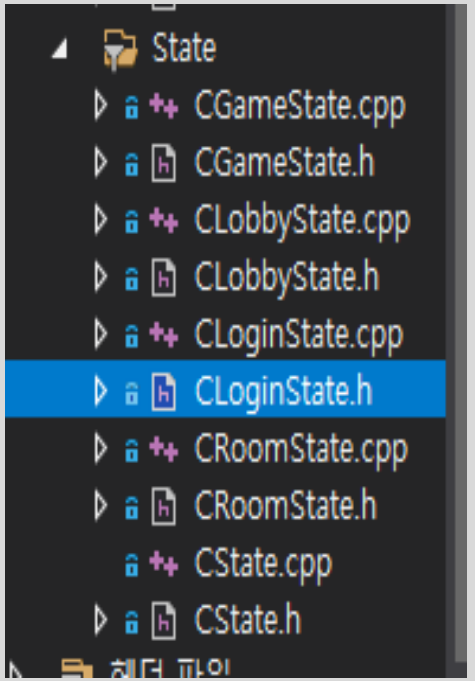
State 패턴을 이용하여
현재 요청 들어온
client(Session)의
현재 상태에 따라
수행할 명령을 다르게 합니다.

IOCP_Server (전역 범위)

```
1   #pragma once
2
3   class CSession;
4
5   class CState
6   {
7   public:
8       CState(CSession* _session) { m_session = _session; }
9       virtual void Recv(t_ThreadInfo* _threadinfo) = 0;
10      virtual void Send(t_ThreadInfo* _threadinfo) = 0;
11  protected:
12      CSession* m_session;
13  };
14
15
```

모두(으)로 이동

도움말을 보려면 ?을(를) 입력하세요.




```
CSector* GetSector() { ... }
CState* GetState() { return m_curstate; }
CState* GetLoginState() { ... }
CState* GetLobbyState() { ... }
CState* GetRoomState() { ... }
CState* GetGameState() { ... }
void SetState(CState* _state) { ... }
```

CSession.h

사용 언어 : C++

Dump

에러 발생에 대한 Dump 파일 생성 및 Text 파일로 필요한 내용 기록하도록 했습니다.

저는 그때의 메모리 상황과 어떤 패킷에 대해 어떤 처리를 하다 오류가 발생 했는지의 정보를 알기 위해 State , Protocol 정보를 남겼습니다.

Exception_Handler.h

```
#pragma once
#include "pch.h"

#pragma warning(push)
#pragma warning(disable : 4091)
#include <DbgHelp.h>
#pragma warning(pop)
#include "CLock.h"
#include "CLockGuard.h"

#pragma comment(lib,"Dbghelp.lib")

class Exception_Handler
{
private:
    static Exception_Handler* volatile m_instance;
    static CLock* m_lock;

    Exception_Handler() {}
    Exception_Handler(const Exception_Handler& other);
    ~Exception_Handler(){}
public:
    static void volatile Create()
    {
        if (m_instance == nullptr)
        {
            CLockGuard<CLock> lock(m_lock);
            if (m_instance == nullptr)
            {
                m_instance = new Exception_Handler();
            }
        }
    }
    static Exception_Handler* volatile Instance()
    {
        return m_instance;
    }
    static void volatile Destroy()
    {
        delete m_instance;
        delete m_lock;
    }

public:
    DWORD _initalize(__in LPCTSTR dump_file_name,
        __in const MINIDUMP_TYPE dump_type = MINIDUMP_TYPE::MiniDumpNormal);
    DWORD run();
    static LONG WINAPI Exception_Callback(__in struct _EXCEPTION_POINTERS* exceptioninfo);

private:
    wstring  _dump_file_name;
    MINIDUMP_TYPE _dump_type;
    LPTOP_LEVEL_EXCEPTION_FILTER _prev_filter;
};
```

[2022_9_4] Log.txt - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
[날짜:2022년9월4일] [시간: 1시 23분]

memory info
dwMemoryLoad: 0
dwTotalPhys: 8364258
dwAvailPhys: 1957472
dwTotalPageFile: 14036016
dwAvilPageFile: 1591052
dwTotalVirtual: 4294967232
dwAvailVirtual: 4292810040



thread info
IO_TYPE: RECV
E_STATE: LOGIN
MAINPROTOCOL:LOGIN
SUBPROTOCOL:LoginInfo
DETAILPROTOCOL:DETAILPROTOCOL 없음
```

server_exception.dmp 2022-09

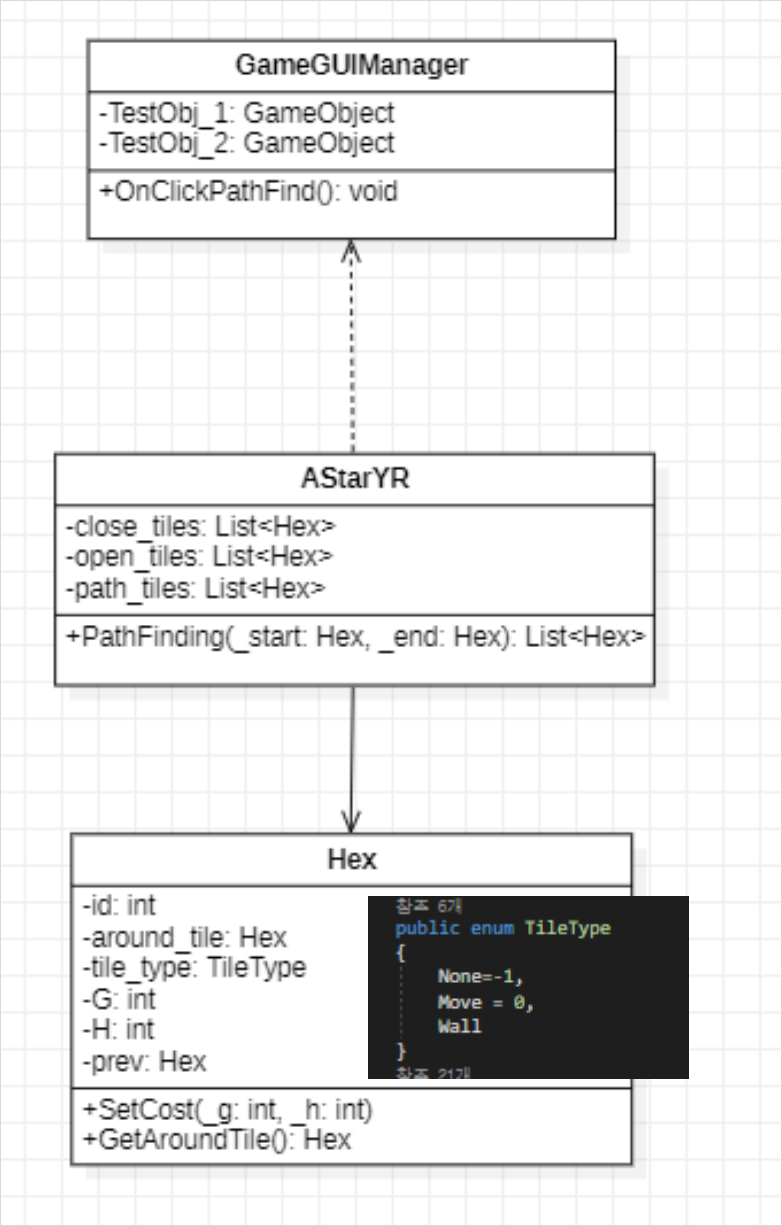
Text log 와 dump 파일

```
LONG __stdcall Exception_Handler::Exception_Callback(_EXCEPTION_POINTERS* exceptioninfo)
{
    do { ... } while (false);

    MEMORYSTATUS memoryinfo;
    GlobalMemoryStatus(&memoryinfo);

    t_ThreadInfo* curthread = CIocp::g_threadinfo[::GetCurrentThreadId()];
    wstring iotype = IOTYPEtoString(curthread->io_type);
    wstring state = ESTATEtoString(curthread->cur_state);
    wstring main, sub, detail;
    PROTOCOLtoString(curthread->protocol, curthread->cur_state,&main, &sub, &detail);
    CLogMgr::GetInst()->
        FileWriteLog(_T("\n\nmemory info\ndwMemoryLoad: %lu\ndwTotalPhys: %lu\ndwAvailPhys: %lu\ndwTotalPageFile: %lu\ndwAvilPageFile: %lu\ndwTotalVirtua
            , memoryinfo.dwMemoryLoad/(1024*2), memoryinfo.dwTotalPhys / (1024 * 1024), memoryinfo.dwAvailPhys / (1024 * 1024), memoryinfo.dwTotalPageFil

    return (Exception_Handler::Instance()->_prev_filter) ? Exception_Handler::Instance()->_prev_filter(exceptioninfo) : EXCEPTION_EXECUTE_HANDLER;
}
```

사용 언어 : C++

플레이어 추적 기능

프로젝트 초기에
전체적인 기획안을 듣고
Boss,Monster가 Player를
추적해야 하는 기능이 필요하다
여겨져 A*를 구현하였습니다.




```cs
public List<Hex> PathFinding(Hex _start, Hex _end)
{
    TileClear();
    start_tile = _start;
    end_tile = _end;
    open_tiles.Add(start_tile);
    Hex current = start_tile;
    start_tile.SetCost(-1, Mathf.Abs(start_tile.GetX - end_tile.GetX) + Mathf.Abs(start_tile.GetY - end_tile.GetY)) ;

    while (open_tiles.Any())
    {
        current = open_tiles[0];
        foreach (Hex t in open_tiles)
        {
            if(t.GetF<current.GetF||t.GetF==current.GetF&&t._H<=current._H)
            {
                current = t;
            }
        }
        if(current.ID==end_tile.ID)
        {
            GetPathList(end_tile);
            return path_tiles;
        }
        close_tiles.Add(current);
        open_tiles.Remove(current);
        SetMovePrice(current);
    }
    return null;

}
```

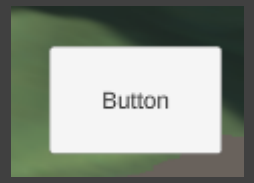
AStartYR.cs

```cs
public void OnClickPathFind()
{
    #region
    move_pos = a_star.PathFinding(start,end);

    foreach (var tile in move_pos)
    {
        MapManager.Instance.DrawLine(tile);
    }
}
```

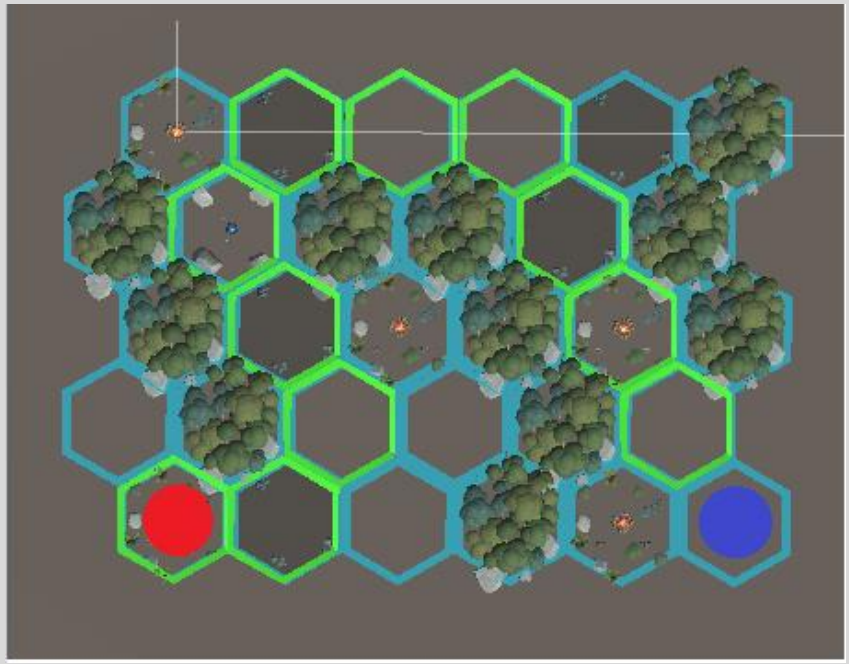
GameGUIManager.cs

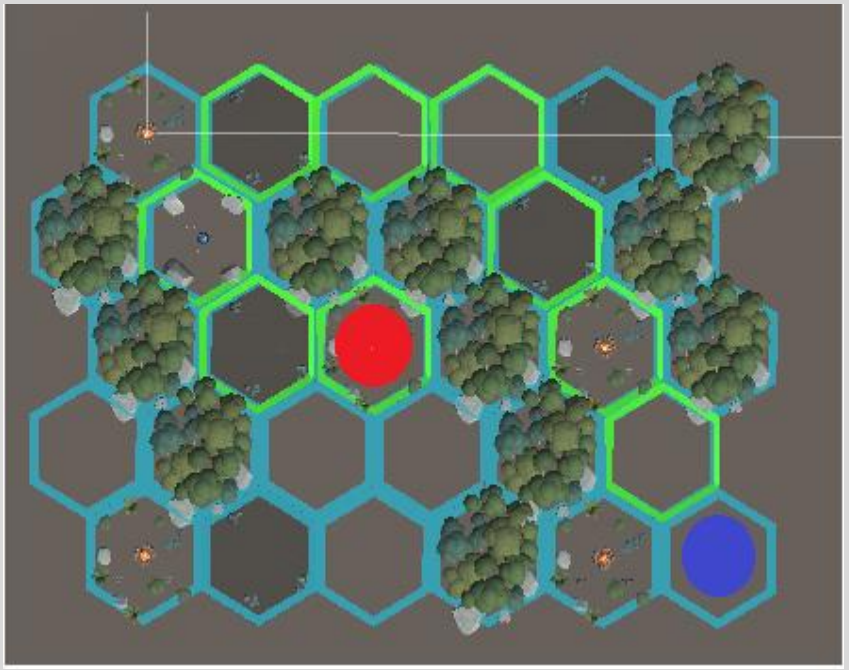
사용 언어 : C#

플레이어 추적 기능



테스트를 위해
버튼을 누르면
출발지에서 목적지까지
의 경로를 계산해서 출
력하게 했습니다.





빨간 원 = 목표 지점
파란 원 = 출발 지점

Excel Reader

기획님께서 주시는

Excel data를
csv 파일과
Script 파일로
자동으로 찍어주도록 하여

팀원들과
함께 사용했습니다.

1.엑셀리더 사용법.docx

사용법을 적어 두어
팀원들이 보고 쓸 수
있도록 하였습니다.

```
namespace make_excel
{

    class Program
    {
        static vo   class make_excel.Program
        {

            // 파일 임포트시 파일경로를 알아온다.
            string filepath =
                @"D:\GitSources\Yggdrasil\ExcelData\Data_Table";

            복사를 위한 테이블 명
            복사를 위한 시트 명

            // 경로를 알면 파일이름을 가져올수 있을듯?
            string filename = "BossSkill.xlsx";



            //Excel.Application ExcelObj = new Excel.Application();
            //Excel.Workbook theWorkbook = null;
            //string strPath = "MENTION PATH OF EXCEL FILE HERE";
            //theWorkbook = ExcelObj.Workbooks.Open(strPath, missing, missing, missing, missing, missing, missing, missing, missing, missing, missing, mi
            //Excel.Sheets sheets = theWorkbook.Worksheets;
            //Excel.Worksheet worksheet = (Excel.Worksheet)sheets.get_Item(1);//Get the reference of second worksheet
            //string strWorksheetName = worksheet.Name;//Get the name of worksheet.

            // 시트 이름 가져오는거 api로 구현되어있음.
            string sheetname = "BossStat_Table";

            if (!filepath.EndsWith("\\"))
                filepath+="\\";



            //ExcelReader를 만드는 부분.
            ExcelReader excel = new ExcelReader();
            excel._Initialize(filepath + filename);
            excel.ReadSheet(sheetname);
            excel._Finalize();
        }
    }


    class ExcelReader
    {
        public struct Pos...

        Application m_app = null;
        Workbook m_workbook = null;
        string filepath;
        bool IsFinalized = false;

        public ExcelReader()...

        ~ExcelReader()...


        public bool _Initialize(string path)...



        public void _Finalize()...
        public void DeleteObj(Object obj)...

        public void ReadSheet(string sheetName)...

        void WriteFile(Excel.Worksheet sheet,int left,int right,int top,int bottom)...

        void CreateFile(string path)...

        void SaveFile(string path,string text)...
    }
}
```

Script 2022-08-17 오후 11:57 파일 폴더
Txt 2022-08-17 오후 11:57 파일 폴더

Map_Table.txt - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
560001`Forest`960001`1`1`2`962001`0`0;
560001`Forest`960001`2`1`4`964001`0`301002
560001`Forest`960001`3`1`0`0`0`0;
560001`Forest`960001`4`1`0`0`0`0;
560001`Forest`960001`5`1`2`964001`0`0;
560001`Forest`960001`6`1`1`961001
560001`Forest`960001`1`2`1`961001
```

Map_Table.txt
Map_Table.txt.meta

CSV

Map_TableExcelLoader.cs

기타 파일 | Map_TableExcel | MapTabl

```
using System.Collections.Generic;
using UnityEngine;

[System.Serializable]
public struct Map_TableExcel
{
    public int MapTableIndex;
    public string StageName;
    public int StgConcept;
    public int StgHorz;
    public int StgVert;
    public int TileType;
    public int TilePrefeb;
    public int BossSummon;
    public int BuffSummon;
}



/*==================================*/

[CreateAssetMenu(fileName="Map_TableLoader", menuName= "Scriptable Object/Map_TableLoader")]
public class Map_TableExcelLoader :ScriptableObject
{
    [SerializeField] string filepath =@"Assets\00.Data\Txt\Map_Table.txt";
    public List<Map_TableExcel> DataList;

    private Map_TableExcel Read(string line)
    {
        line = line.TrimStart('\n');

        Map_TableExcel data = new Map_TableExcel();
        int idx =0;
        string[] strs= line.Split('`');

        data.MapTableIndex = int.Parse(strs[idx++]);
        data.StageName = strs[idx++];
        data.StgConcept = int.Parse(strs[idx++]);
        data.StgHorz = int.Parse(strs[idx++]);
        data.StgVert = int.Parse(strs[idx++]);
        data.TileType = int.Parse(strs[idx++]);
        data.TilePrefeb = int.Parse(strs[idx++]);
        data.BossSummon = int.Parse(strs[idx++]);
        data.BuffSummon = int.Parse(strs[idx++]);

        return data;
    }
    [ContextMenu("파일 읽기")]
    public void ReadAllFile()
    {
        DataList=new List<Map_TableExcel>();

        string currentpath = System.IO.Directory.GetCurrentDirectory();
        string allText = System.IO.File.ReadAllText(System.IO.Path.Combine(currentpath,filepath));
        string[] strs = allText.Split(';');

        foreach (var item in strs)
        {
            if(item.Length<2)
                continue;
            Map_TableExcel data = Read(item);
            DataList.Add(data);
        }
    }
}
```

Script

Map_TableExcelLoader.cs
Map_TableExcelLoader.cs.meta

사용 언어 : C#

Mouse Manager

해당 기능 프로젝트 사용 X

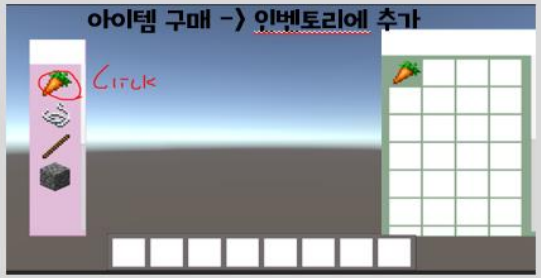
Mouse event를
직접 구현해서 써보고
싶다는 생각에 공부 목적
으로 만들었습니다.

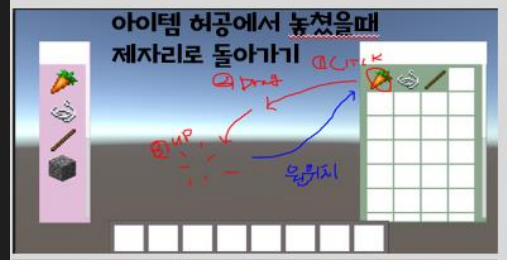
해당 기능을 토대로
임의의 인벤토리 기능을
만들어 사용하였습니다.

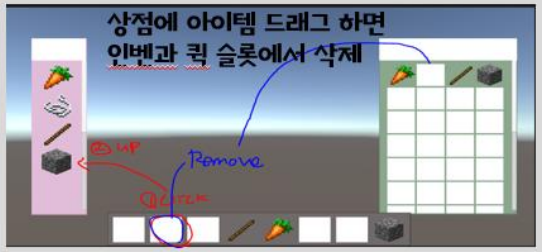
MouseManager.cs

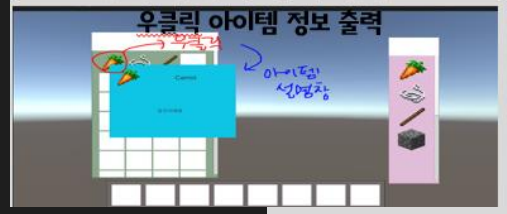
기타 파일 | TAG_TYPE | NONE

```
using UnityEngine.EventSystems;
using UnityEngine.UI;
using System;

//왼,오 다운,드래그 만 구현됨.
public enum TAG_TYPE
{
    NONE,
    MenuStart,
    MAX
}
public enum MOUSE_TYPE
{
    NONE,
    LEFTDOWN_BTN,
    LEFTUP_BTN,
    LEFTDRAG_BTN,
    RIGHTDOWN_BTN,
    RIGHTUP_BTN,
    MAX
}
// 이벤트 호출시 넘겨줄 마우스 정보 구조체.
public struct MouseArgs
{
    public Vector2 beforPos;
    public Vector2 currentPos;
    public GameObject beforGameObject;
    public TAG_TYPE befortag;
    public bool is_LeftDown;
    public bool is_RightDown;
}
public class MouseManager : Singleton_Ver2.Singleton<MouseManager>
{
    [SerializeField]
    Canvas m_canvas;
    GraphicRaycaster m_Raycaster;
    PointerEventData m_PointerEventData;
    EventSystem m_EventSystem;

    public delegate void MouseEventHandler(GameObject _selected, MouseArgs args);

    Dictionary<MOUSE_TYPE, Dictionary<TAG_TYPE, List<MouseEventHandler>>> MouseEvents;

    Dictionary<MOUSE_TYPE, List<TAG_TYPE>> MouseTagList;

    GameObject m_currentGameObject;
    MouseArgs m_current_args;
    float current_mousedown_time;

    TAG_TYPE m_currenttag;
    bool m_isFind;
    object temptag;
    //마우스 이벤트 등록
    public void MouseEvent_Register(MOUSE_TYPE p_mouse, TAG_TYPE p_tag, MouseEventHandler p_event)...

    public Vector3 MousePos...
    public void Init()...
    private void Awake()...
    void Start()...
    private void Update()...
}
```









사용 언어 : C#

감사합니다!