이문구

21.2. 포트폴리오 기술문서

Order Of Ecclesia 모작

플레이 영상:
https://www.youtube.com/watch?v=JglkA2rmXec&t=8628s
장르 : 액션 플랫폼
제작 영상:
https://youtu.be/-GFNKDv2yAw
깃허브 주소:
https://github.com/LeeMungu/Order-Of-Ecclesia-Imitation
인원 : 1인(이문구)
시간 : 21.01.23 ~ 21.02.07

구조도


MainGame
Player
Monster Manager
Object Manage
Effect Manager
Map
Weapon
Monster
Object
Effect
- Bat
- Zombi
- Boss
- Wall
- Bottom
- Door
- HitEffect
- Number
-MapEffect

주요 포인트

상태 전환에 따른 부드러운 모션

아이템 습득에 따른 무기 전환

매니저 클래스에 의한 객체 관리

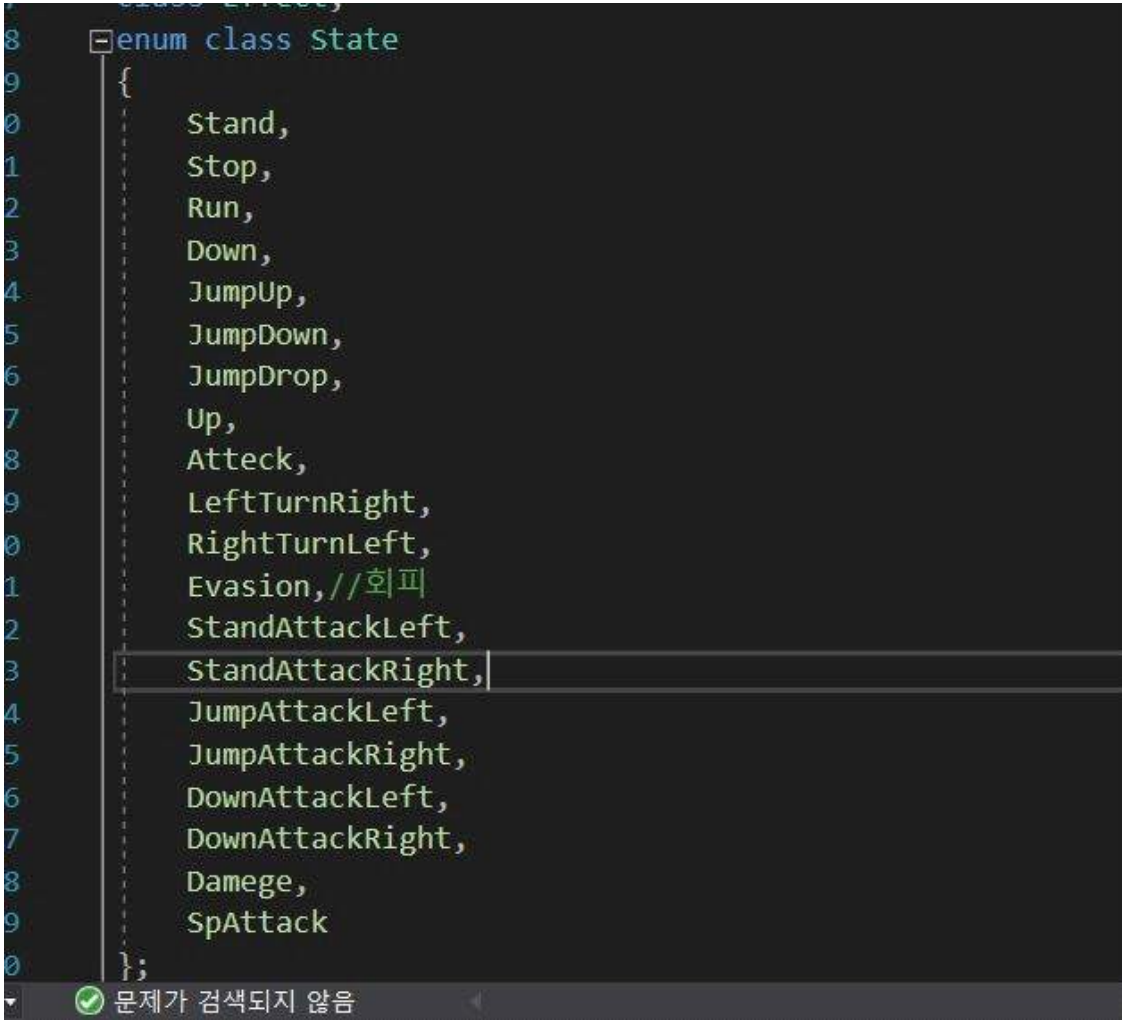
카메라 모션 및 맵 구현

상태 전환에 따른 부드러운 모션

enumClass를 활용한 다양한 상태 처리

특정 모션이 나오려면 어떤 상황인지에 따라 여부를 확인하고 처리해야 했습니다. 처음에는 bool변수로 모션 여부를 체크했습니다 .

하지만 상태가 너무 많고 상태여부를 타 클래스에서 모션에 따른 기타 처리가 필요한경우를 생각해 enum Class를 사용 하였습니다 .



아이템 습득에 따른 무기 전환

Item-Effect-Player를 이동한 Item정보

장비 습득 구조가 특이한 게임으로 공중에 떠있는 문양근처에서 Up모션을 취했을 때 문양 이미지가 빨려들어가면서 습득됨

위치에 따라 다른 이미지를 가져오는 것을 구현하기 힘들었습니다.

본 게임 영상

모작 게임 영상

그래서 문양 근처에서 Up모션을 취했을때 특정 Effect가 생성되서 해당 Item이 Effect를 통해서 Player에 습득되는 형태를 취했습니다 .

아이템 습득에 따른 무기 전환

Item-Effect-Player를 이동한 Item정보

Item정보가 Effect를 지나서 Player에 들어갈 수 있도록 enumClass를 이용하여 장비 타입을 구현하였습니다 .

ItemManager에서 해당 이펙트와 충돌하면 아이템이 가지고 있는 enumClass를 Player에게 넘겨주는 처리를 하였습니다.

Item.h

```h
#pragma once
#include "GameObject.h"
#include "Image.h"

enum class WeaponType
{
    Confodere,//레이피어
    Secare//검
};
```

ItemManager.cpp

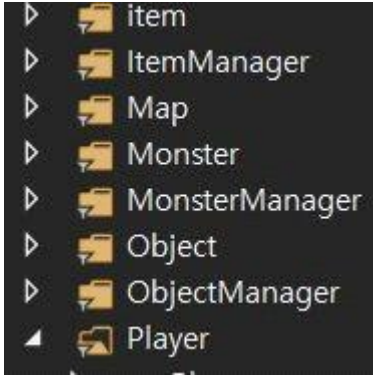
```cpp
//충돌후 제거
if (mItemList.size() != NULL)
{
    for (int i = 0; i < mItemList.size(); i++)
    {
        if (mItemList[i]->GetAppear2() == true)
        {
            //플레이어에 해당 item의 WeaponType을 넣어준다.
            mPlayer->SetWeaponType(mItemList[i]->GetWeaponType());
            mItemList[i]->Release();
            SafeDelete(mItemList[i]);
            mItemList.erase(mItemList.begin() + i);
            i--;
        }
    }
```

매니저 클래스를 통한 객체 관리

Sin넘어감에 따른 Object나 Monster, Item등 의 관리 및 충돌 처리

여러 sin이 이어져있는 구조의 게임으로 각 sin마다의 정보를 저장하고 여러 객체들을 관리할 클래스의 필요성을 느꼈습니다.

그래서 매니저 클래스를 만들어서 Sin의 정보를 저장하고 Init함수에 매개변수를 두어서 특정 sin에 특정 자료 들이 나오게 만들었습니다 .



Item, Monster, Object 들의 매니저

```
mPlayer->Update();
mMap->SetPlayer(mPlayer);
mMap->Update();
//스테이지 번호
mStageNum = mPlayer->GetStageNum();
//스테이지 이동 여부
mStageSet = mPlayer->GetStageSet();
if (mStageSet == false)
{
    mObjectManager->SetPlayerPtr(mPlayer);
    mObjectManager->Update();

    mMonsterManager->SetPlayerPtr(mPlayer);
    mMonsterManager->Update(mStageNum);

    mItemManager->SetPlayerPtr(mPlayer);
    mItemManager->Update();
}
```

MainGame에서 해당 매니저들이 매개 변수 (스테이지 번호)를 받음

카메라 모션 및 맵구현

Player이동에 따른 카메라 좌표 변화와 각 객체들의 상대 좌표 구현

넓은 맵을 표현하기 위해 Player이동에 따른 카메라 변화를 주어야 했습니다.

그래서 Player의 좌표를 기준으로 Player의 Update전후의 차이 값을 가지고 카메라의 좌표를 구했습니다.

```
void Player::Update()
{
    mBeforeX = mX;
    mBeforeY = mY;
```

업데이트 앞에서 값을 가지고 있다가

```
if ((mX > WINSIZEX / 2 &&
    mObjectList[findNum]->GetRect().right-mX >= WINSIZEX/2))
{
    mCameraX += mBeforeX - mX;
}

if (mStageNum == 1)
{
    mCameraY += mBeforeY - mY;
}
```

상황에 맞춰 후반에 이동 차이를 카메라 좌표값에 더해준다.

카메라 모션 및 맵구현

Player이동에 따른 카메라 좌표 변화와 각 객체들의 상대 좌표 구현

camera의 상대좌표를 구했지만 player의 맴버 변수라서 일일히 빼서 각 클래스들이 가져가는게 문제였습니다.

그래서 매번 Get,Set으로 넣어주지 않게 앞서 구한 카메라 좌표를 각 객체들의 Render함수의 매개변수로 받아서 카메라를 따라가게 만들었습니다 .

```
void Weapon::Render(HDC hdc, float cameraX, float cameraY)
{
    if (mAppearance == true)
    {
        mImage->FrameRender(hdc, mImageX + cameraX, mImageY + cameraY, mIndexX, mIndexY);
    }
```

각 클래스는 카메라 좌표를 Render매개변수를 받아서 보정해서 그려준다.

```
void ObjectManager::Render(HDC hdc,float cameraX, float cameraY)
{
    for (int i = 0; i < mObjectList.size(); ++i)
    {
        mObjectList[i]->Render(hdc,cameraX,cameraY);
```

각 클래스를 관리해주는 매니저 클래스 또한 다음과 같이 매개변수를 넣는다.

```
void MainGame::Render(HDC hdc)
{
    HDC backDC = mBackBuffer->GetHDC();
    PatBlt(backDC, 0, 0, WINSIZEX, WINSIZEY, WHITENESS);
    {
        float cameraX = mPlayer->GetCameraX();
        float cameraY = mPlayer->GetCameraY();
        mBackGroundManager->Render(backDC, cameraX, cameraY);
        mObjectManager->Render(backDC, cameraX, cameraY);
        mPlayer->Render(backDC);
        mMonsterManager->Render(backDC, cameraX, cameraY);
        mItemManager->Render(backDC, cameraX, cameraY);
        mMap->Render(backDC);
```

매니저의 매개변수들은 그림과 같이 MainGame의 Render에서 player로 넣어준다.

감사합니다.

이문구