2013180021

원주연

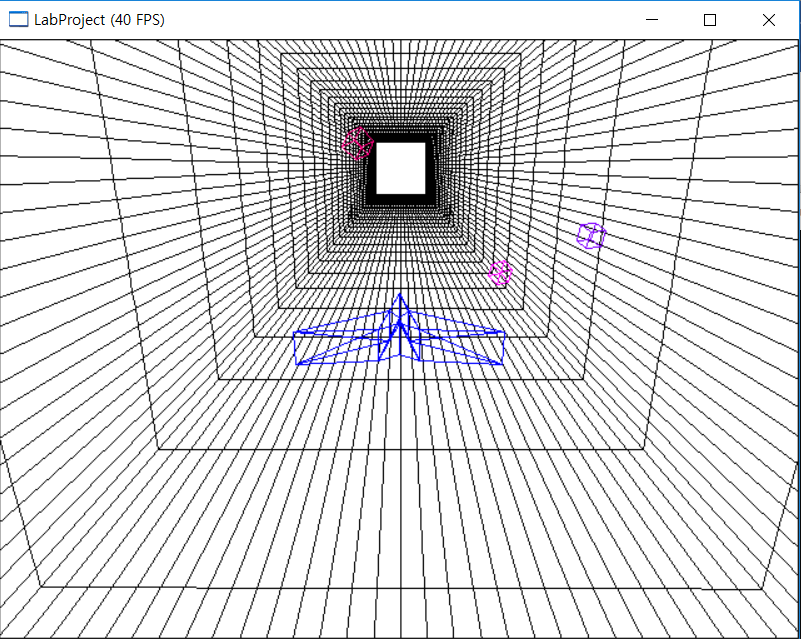
3D 과제 01 Document

**[과제에 대한 목표]**

Direct3D12를 사용하는데 필요한 개념에 익숙해지기 위해 API환경에서 3D환경 프로그램을 구현한다. 과제의 구현 내용을 최대한 구현하는 것이 목표이다.

**[구현한 내용]**

e-class에 올라온 explosion 샘플 프로젝트를 기반으로 프로그램을 구현하였다.



기본으로 생성되 있던 10개의 적들은 삭제를 하였고 시간마다 적을 생성하는 void CScene::CreateObjectPerTime(float fElapsedTime)함수를 정의하여 적들을 생성한다.

빨간색을 제외한 적들을 1초마다 생성하기 위해 m\_fCreateTimes += fElapsedTime;로

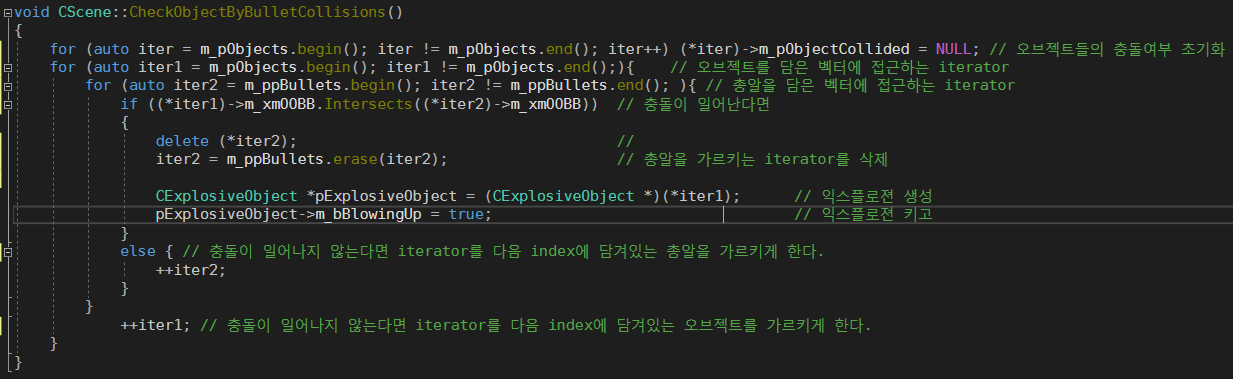
m\_fCreateTimes가 1.0f가 넘으면 Blue, Green, Pink 적 중 하나를 랜덤으로 골라 생성한다.

m\_fRCreateTimes >= 10.0f 앞의 방식과 똑같이 10초마다 적을 생성하기 위해 같은 방식을 사용하였다.

적들은 시간에 따라 계속 생성되고 플레이어가 쏜 미사일과 충돌하면 없어지길 반복 해야 되기 때문에 정적인 배열보다 동적 자료형이 적합하다 생각하여 벡터 Container에 적의 객체를 담았다.

vector<CGameObject\*> m\_pObjects; 적들과 마찬가지로 미사일도 삭제와 생성을 반복해야 해서 이와 같은 방법으로 벡터에 미사일 객체를 담았다. vector<CGameObject\*> m\_ppBullets;

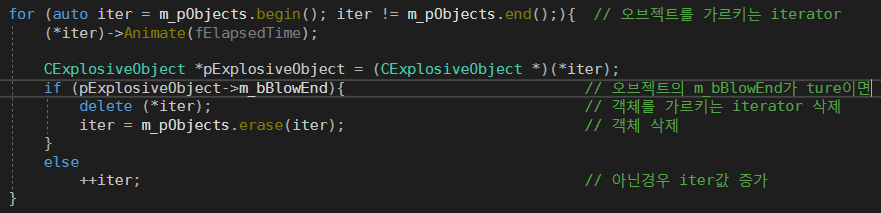
총알과 오브젝트의 충돌을 처리하기 위해 void CScene::CheckObjectByBulletCollisions()함수를 정의하여 처리하였다. 총알과 오브젝트 둘다 vector container에 담아서 iterator로 접근을 하였다.



처음에는 이 함수 내에서 오브젝트를 가르키는 iterator도 삭제를 해주었는데 그렇게 하였더니 파티클이 터져 그려지기 전에 오브젝트가 삭제되었다. 파티클이 터지는 것을 다 그려주고 객체를 삭제하기 위해 bool m\_bBlowEnd를 선언하여 파티클이 다 그려지고 나서 값을 true로 설정해 주었다.

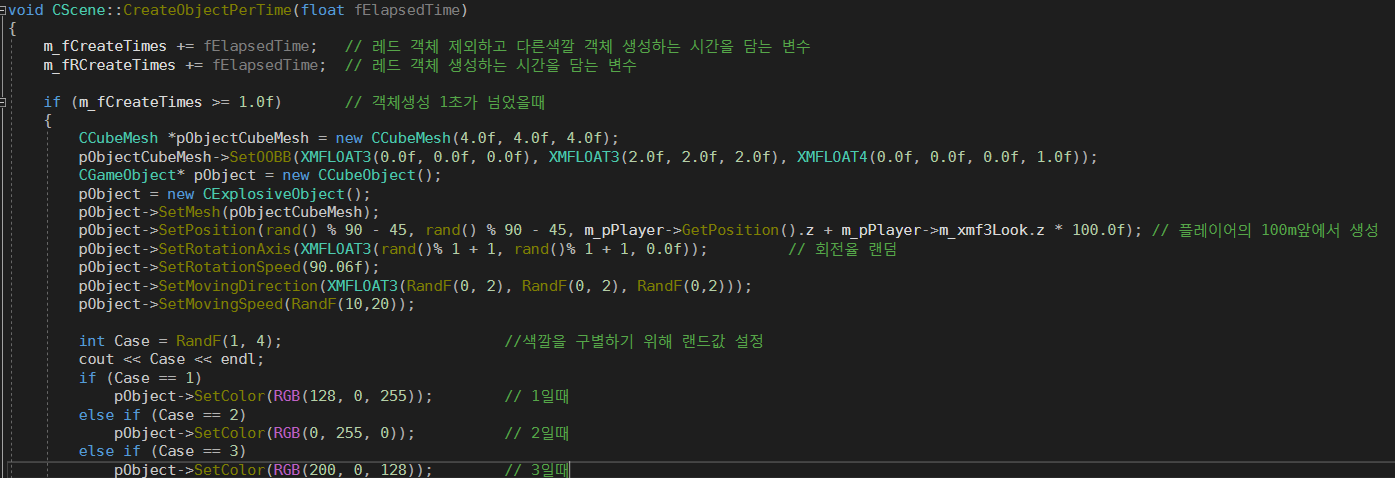
if (m\_fElapsedTimes <= m\_fDuration){ ...} else{ m\_bBlowEnd = true; … }

그리고 CScene::Animate함수에서 삭제를 해 주었다.



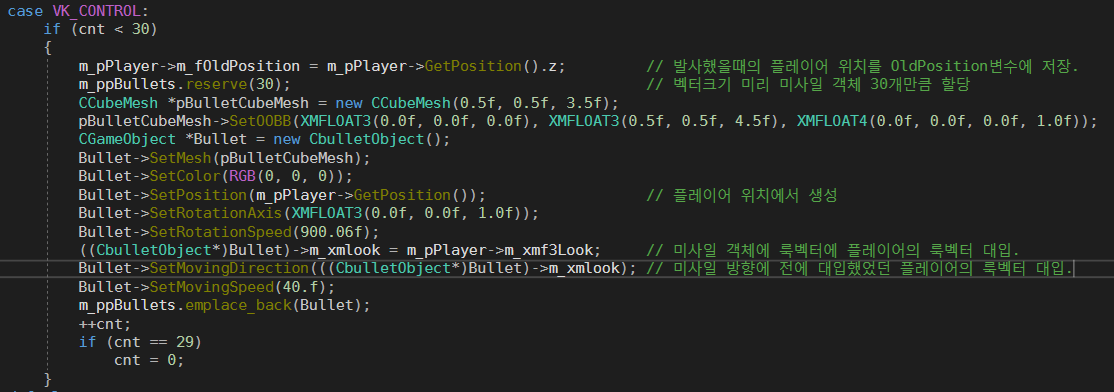
여기서 처음에 for문에 iter++ 증감식을 넣어줬었다. If문 조건에 충족해 그 객체를 지울 경우 iter가 무효화 되어서 오류가 많이 나서 여기서 for문 구조를 전체적으로 한번 싹 다 바꾸었다.

시간마다 객체를 생성하기 위해서 void CScene::CreateObjectPerTime(float fElapsedTime) 인자값으로 시간을 받아와 객체를 생성하였다.



플레이어 100미터 앞에서 생성하기 위해서 현재 플레이어의 z축 좌표를 얻어와 거기서 플레이어의 룩 벡터에 100m를 더하여 생성하였다. 밑에서 Case변수를 선언하여 RandF()함수를 돌려 케이스에 따라 다른 색상의 오브젝트가 생성되게 만들었다. 그 생성한 객체를 백터에 push\_back 하여 Animate에서 새로 생성된 객체가 그려질 수 있게 하였다. 생성한 후 m\_fCreateTimes의 값을 0으로 초기화하여 다시 1초마다 if문 안에 들어올 수 있도록 하였다.

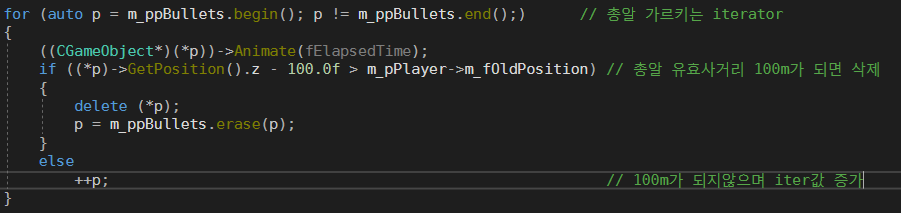
그 밑에는 if (m\_fRCreateTimes >= 10.0f) if구문으로 10초마다 들어오게 하여 위와 동일한 방식으로 크기가 조금 더 큰 빨간색 오브젝트를 생성하여 벡터에 넣었다.



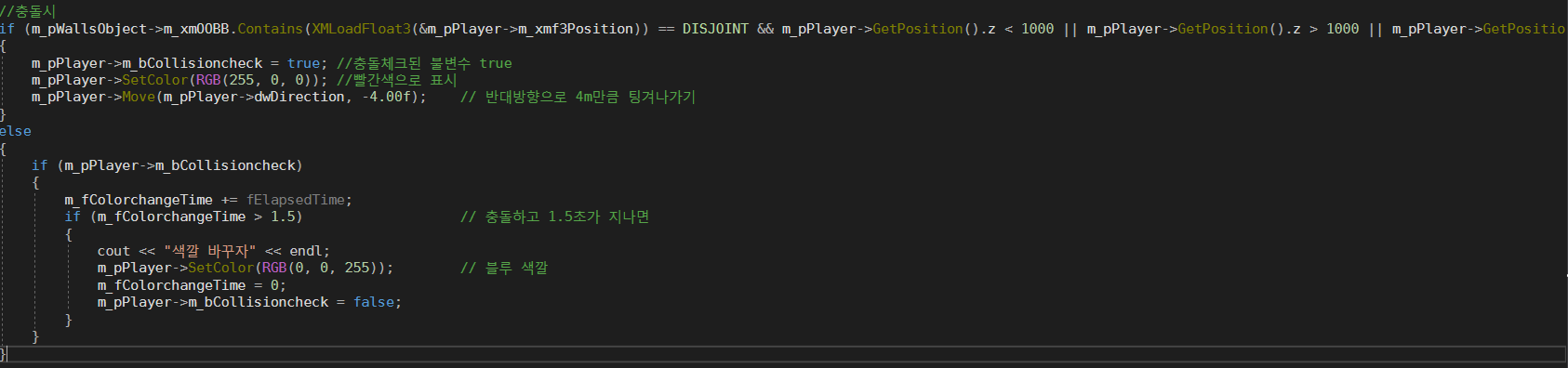
컨트롤을 눌렀을 때 플레이어의 현재 좌표를 저장하는 변수를 하나 선언해서 플레이어의 z값을 저장한다. 이 oldposition 변수는 총알의 유효사거리 100m를 구현하기 위한 변수이다. 처음에 총알의 z값 – 플레이어의 좌표값의 절대값이 100이 넘으면 삭제하도록 하였는데, 플레이어가 앞으로 이동하면서 총알을 쏘면 총알의 유효사거리가 더 늘어나는 경우가 생겨 이 변수를 두어 유효사거리를 구현하였다.

XMFLOAT3 m\_xmlook 변수는 CGameObject의 상속을 받는 총알 객체에 멤버변수이다. 이것도 위의 oldPosition의 변수와 비슷한 이유로 추가를 하게 되었다.

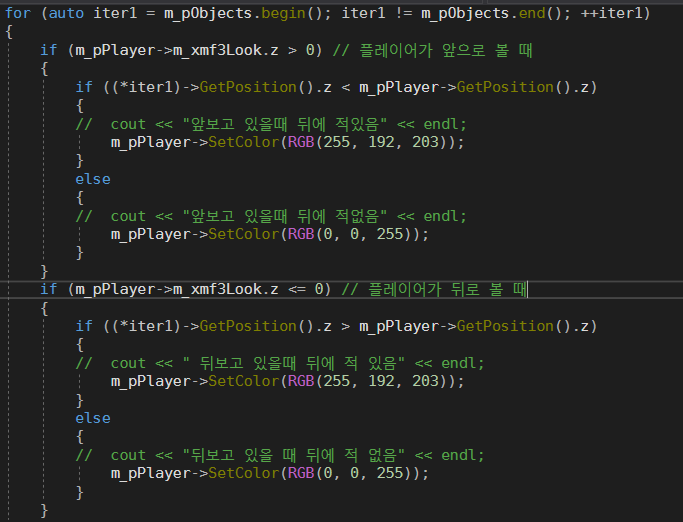
처음에 Bullet>SetMovingDirection(m\_pPlayer->m\_xmf3look)으로 총알의 방향을 설정해 주었다. 이렇게 하니 총알이 날라가는 도중에 플레이어가 방향을 틀면 총알도 같이 방향을 트는 상황이 발생하여 총알발사(Ctrl 키)를 눌렀을 때의 플레이어 룩 벡터를 저장하여 그 방향으로 쏘는 방식으로 구현하였다.



미사일 삭제는 Animate함수에서 처리를 하였다.



플레이어가 벽에 충돌했을 시 충돌을 했는지 나타내는 bool 변수 m\_bCollisioncheck = true로 바꾸고 색깔을 바꾼다. Bool 변수가 true일 때 시간을 재서 1.5초가 넘으면 원래 색으로 돌아오게 만들었다.



적이 플레이어 뒤에 있는지 판단하는 것은 내가 일단 앞을 보고 있는지 뒤를 보고있는지는 가장 먼저 내가 앞을 보고 있는지 뒤를 보고있는지 룩벡터의 z값으로 구분을 한 뒤 오브젝트와 플레이어의 위치 z값을 비교하여 처리하였다.

**[조작법]**

움직이기 : 화살표 방향키

총알쏘기 : Ctrl키

**[구현하지 못한 내용]**

구현하지 못한 내용은 보스와 아이템 그리고 피킹이다.

객체의 생성과 소멸을 자주 반복 하다 보니 아직 사용이 익숙치 못한 iterator에 의해서 프로그램이 터지는 경우가 많아 버그를 잡는데 많은 시간이 소모되었다. 아이템이나 보스 같은 경우 충분히 생각을 해보고 구현을 하려고 했으나 버그를 잡는데 시간을 많이 소모하여 시간상의 문제로 구현하지 못했다. 아이템의 경우 빨간색 오브젝트만 아이템을 뿌릴 수 있다. 그래서 Object 클래스의 멤버 변수로 빨간색 객체를 판단할 수 있는 bool변수를 두어 빨간색 객체를 생성할 때 true로 설정하여 생성을 한다. 미사일과 오브젝트가 충돌했을 시 그 오브젝트의 변수가 true이면 아이템을 생성하는 방식으로 알고리즘을 생각하였었다. 그래서 특수한 키를 입력 받아 아이템이 사용되어지면 객체들의 포지션벡터에서 나의 포지션벡터를 빼서 일정 거리 안에 있으면 그 오브젝트를 삭제시키는 방향으로 생각을 했었다.