**컴퓨터 그래픽스 결과보고서**

**2010180020 박성경**

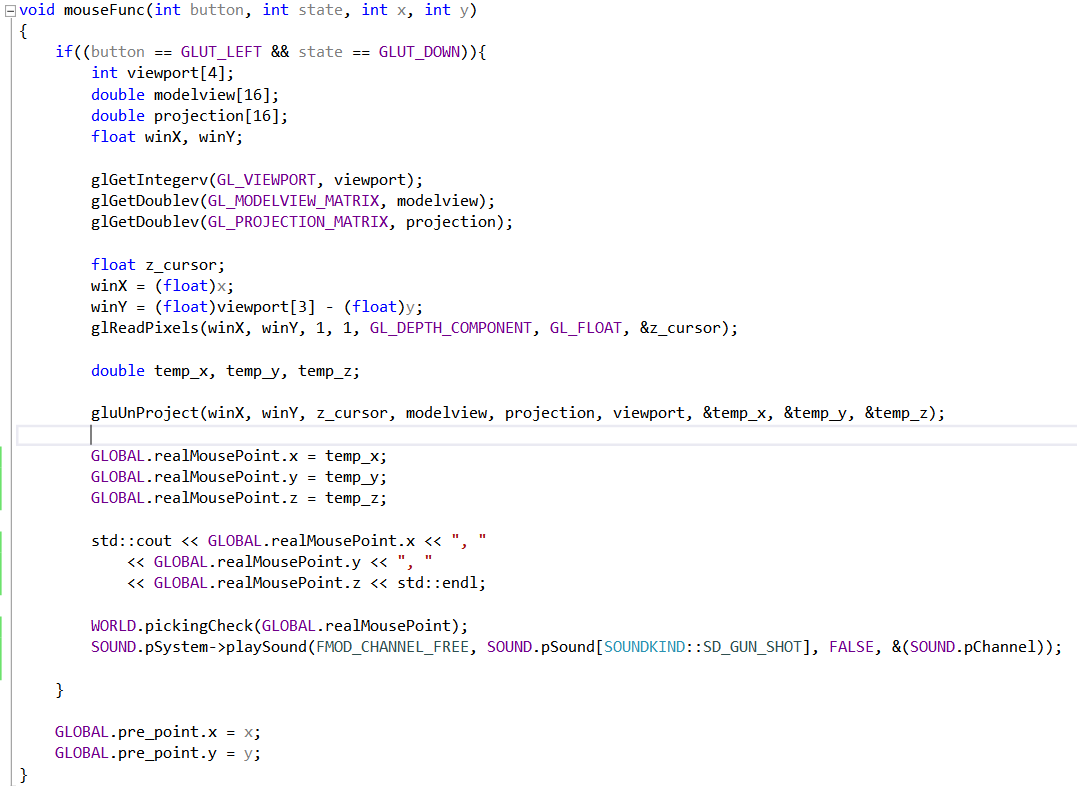
**목차**

**3D Picking --------------------------------------------- 2**

**Particle --------------------------------------------- 5**

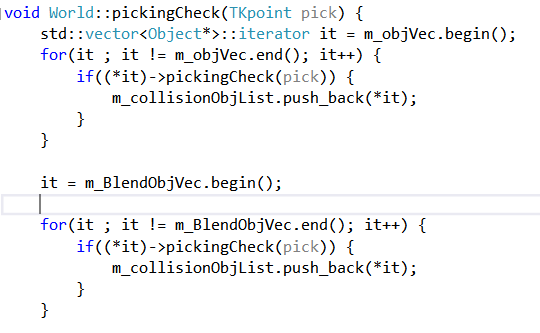
**Camera --------------------------------------------- 7**

**3D Picking**

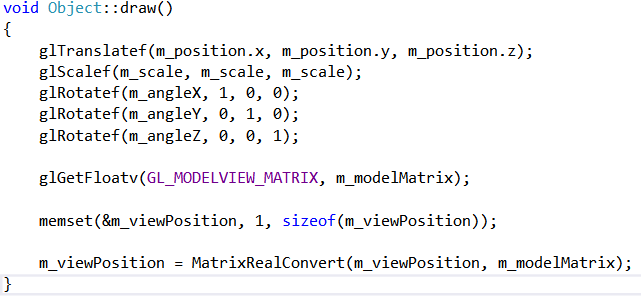


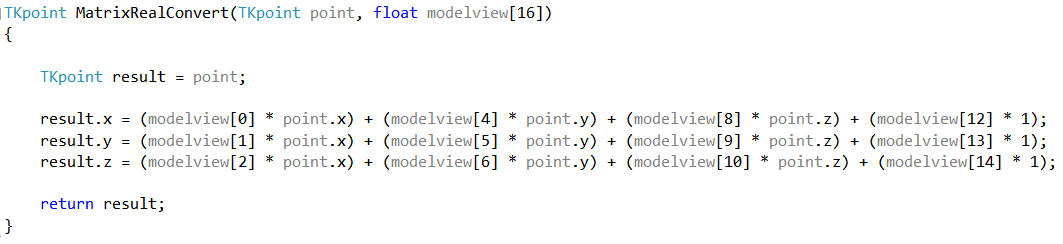
glReadPixels로 현재 픽셀에 대한 z깊이를 얻어옵니다.

gluUnProject함수로 현재 찍은 위치의 실제 좌표값을 얻어옵니다.

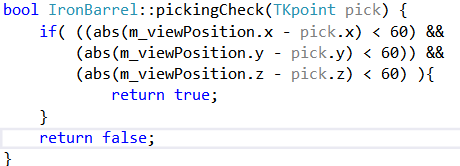


그런 뒤 객체를 관리해주는 World클래스에서 현재 찍은 실제 좌표와 충돌체크가 되는지 확인하고 충돌체크된 오브젝트 리스트 컨테이너에 넣어줍니다.

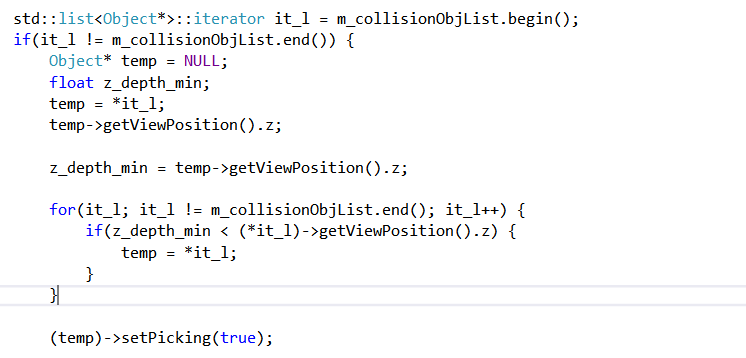




각 객체마다 자신의 화면상 그려지는 실제 좌표를 가지고 있습니다.

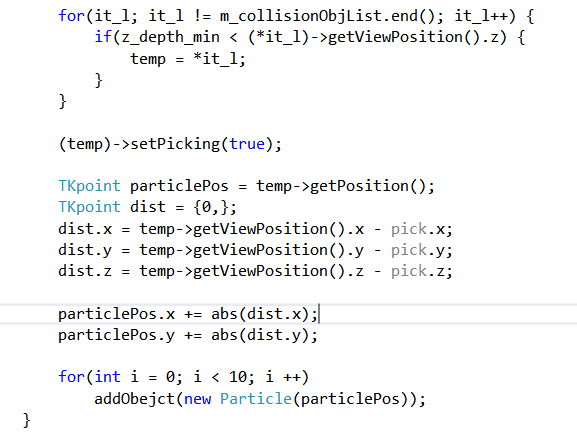


그려지는 실제 좌표를 이용하여 현재 마우스 클릭한 곳과 각 객체마다의 충돌체크 기준으로 체크합니다. 때문에 x축으로 회전시켜 픽킹 할 경우 제대로 픽킹을 검사 못합니다. 하지만 이번 건슈팅 게임에서는 위에서 볼 일은 없어서 일단 불완전한 픽킹을 썼습니다.

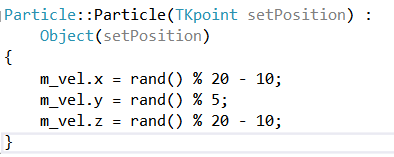


충돌체크된 오브젝트 리스트를 통해 각 viewPosition의 z값에 따라 제일 앞에 있는 객체에게 픽킹 되었다는 것을 알려줍니다. 그 뒤 픽킹이 되었을 때 처리해줄 루틴을 각 객체의 update에서 해주거나 처리하지 않으면 그냥 장애물 객체가 됩니다.

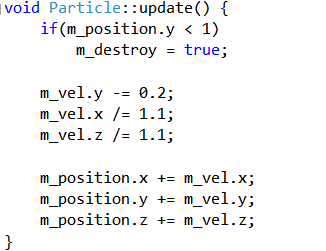
**Particle**



파티클은 이전 픽킹 충돌체크에서 한 물체를 기준으로 약 10개의 큐브 파티클을 뿌려줍니다.



파티클 생성시 가속도값을 랜덤하게 설정하여

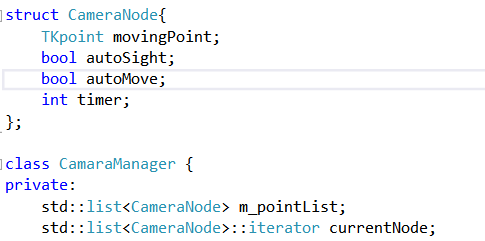


해당 값을 현재 좌표에 더해줍니다.

y값은 점점 아래로 빠르게 내려가게.

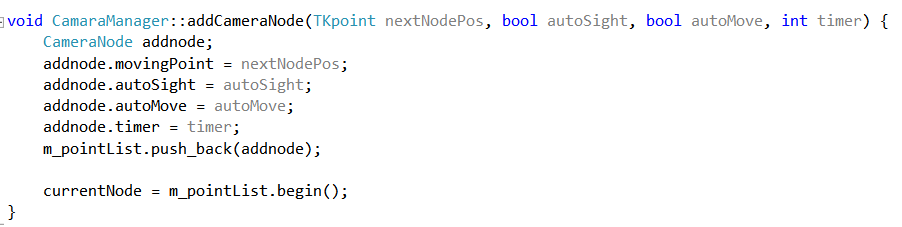
x, z 값은 점점 줄어들어 0으로 수렴하게 하여 포물선으로 뿌려집니다.

**Camera**

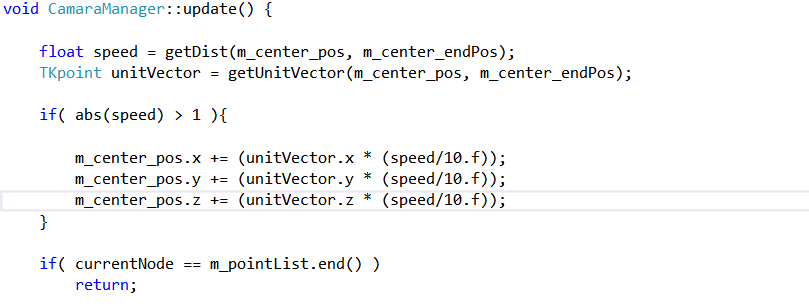


카메라 매니저 클래스는 싱글톤 객체입니다. 다음 지점의 좌표와 시야를 가지고 갈 것인지,

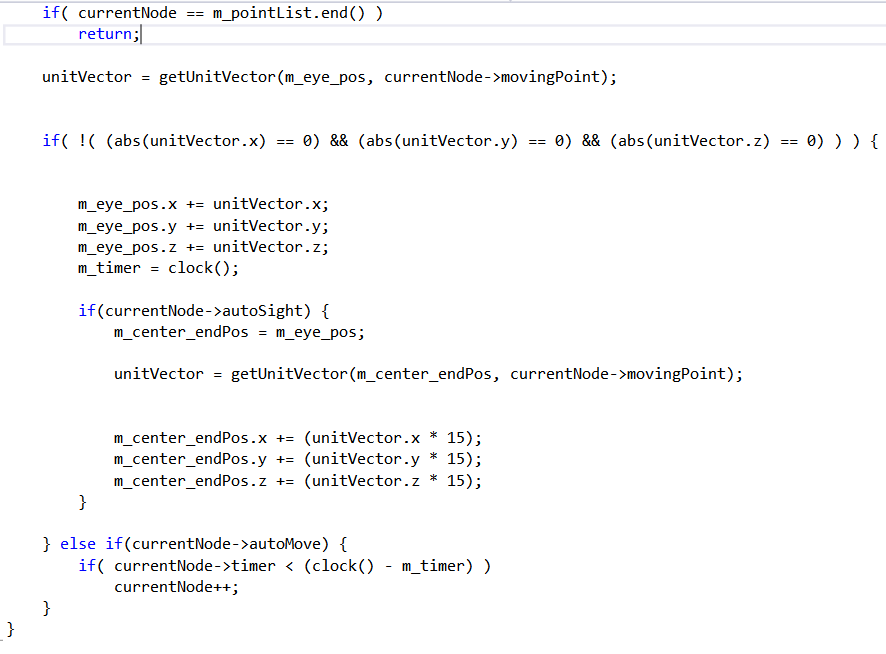
다음으로 자동으로 이동할 것인지, 몇초 뒤에 다음 노드로 이동할 것인지에 대한 정보 구조체로 이루어진 list 컨테이너를 가지고 있습니다.



함수를 호출하여 다음 노드에 대한 정보를 list에 담아줍니다.



현재 바라보고 있는 곳을 내가 바라보고 싶은 좌표인 m\_center\_endPos로 부드럽게 움직여 줍니다. 그리고 현재 노드가 마지막 노드이면 이후 루틴은 실행 하지 않습니다.



현재 노드가 마지막 노드가 아닐시에

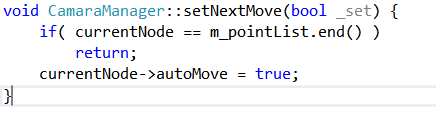
단위벡터로 거리 1만큼씩 카메라를 목표한 지점으로 옮겨줍니다.

거리가 1 이하면 유닛벡터에 대해서 0을 반환하여 오류나는 계산을 방지해주었습니다.

autoSight가 켜져 있는 노드이면 현재 eye의 좌표에서 단위벡터 \* 15만큼 한 거리에 center를 위치 시킵니다. 앞으로 가는 것 같은 효과를 줍니다.

autoSight가 꺼져있으면 최근 center 좌표에 고정되어 시선을 고정 시켜주는 효과를 줍니다.

autoMove가 켜져있으면 자동적으로 다음으로 넘어가거나 정해준 Timer만큼 기다렸다가 다음 노드로 넘어갑니다.



autoMove가 꺼져 있을 시 외부에서 setNextMove를 써주어 다음으로 넘어가도록 설정 해 줍니다.