|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **주차** | 18 주차 | **기간** | 2020.4.27 ~ 2020.05.03 | **지도교수 이용희** | (서명) |
| 이번주 한일 요약 | -행동트리 기초작업  -락 온 기능 추가  -비행기 스피드 표시 | | | | |

<상세 수행내용>

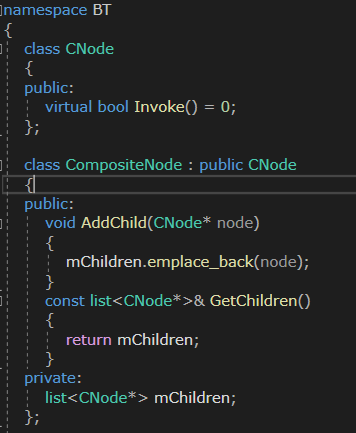
깃 주소 : <https://github.com/zzcx88/DKS_Project.git>

**이재원 :**

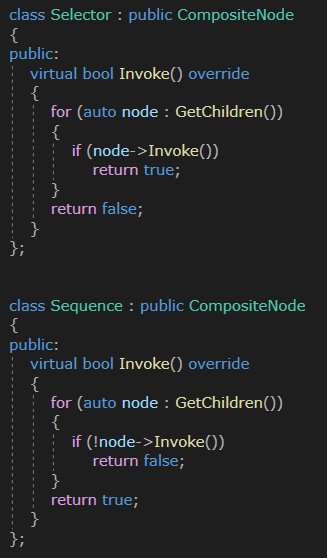
**행동트리를 정의하기 위한 기초 작업을 하였다.**

**1. CNode 클래스 작성**

행동 트리의 기본 노드가 되는 CNode클래스를 작성한다.



CNode는 행위가 성공했는지 못했는지를 판단하는데 필요한 invoke함수와 CNode를 상속받아 작성된 CompositeNode는 자식 노드들을 저장하기 위한 리스트와 자식 노드를 추가, 불러오기 위한 함수를 정의한다.

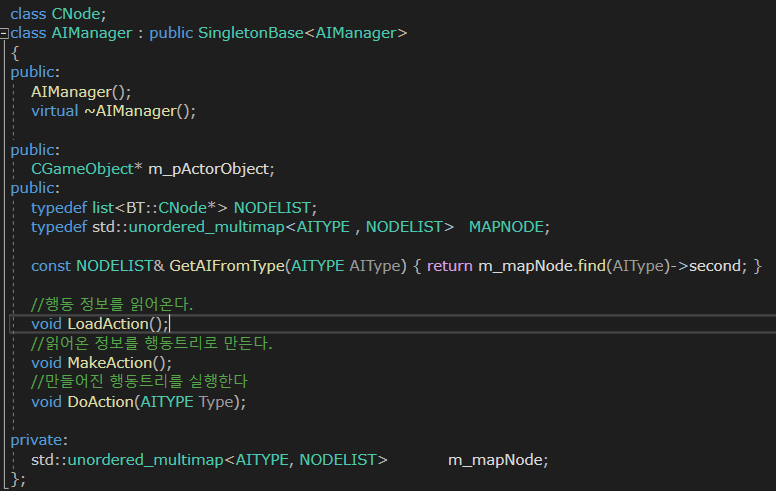


어떤 행위를 실행 할 것인지 결정하기 위한 SelectorNode와 행위를 모두 실행해야 True가 반환되어 SelectorNode의 결정에 영향을 주는 SequenceNode를 정의한다.

SequenceNode노드는 모든 자식 노드가 True를 반환해야 본 노드도 True를 반환하기 때문에 하나만 True여도 True를 반환하는 SelectorNode와 Invoke함수가 차이를 둔다.

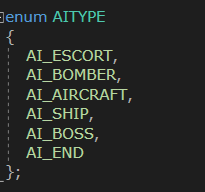
**2. AIManager 클래스 작성**

추후 행동트리 툴에서 만들어진 행위에 대한 정보를 읽어와 행동 트리를 구성하고 게임 오브젝트의 Animate함수에서 호출하여 행동을 적용할 함수를 작성한다.



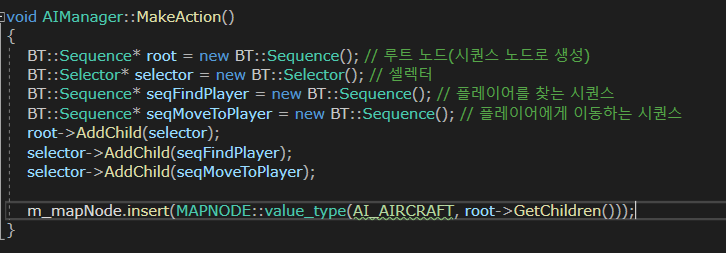
AIManager클래스는 하나의 행동 트리를 정의한 노드가 담긴 리스트를 AITYPE을 키로 가지고 있는 맵 자료 구조를 가지고 있으며 게임 오브젝트는 이 리스트의 노드를 순회하여 행위를 실행한다.

물론 게임 오브젝트는 DoAction 함수를 통해 행위를 호출하며 필요에 따라 GetAIFromType이라는 함수를 통해 직접 맵 노드에 접근할 수도 있다.



AITYPE은 enum으로 정의되어 있으며 하나의 행동트리는 하나의 AITYPE에 대응된다.

따라서 필요한 행동트리의 수만큼 AITYPE의 enum값도 추가 혹은 삭제가 될것이다.



임시로 노드를 구성하여 MakeAction함수를 채워보았다.

노드만 구성하였고 노드 내 행위는 구성하지 않았음으로 아무 동작도 하지 않을 것이지만 우선 트리를 어떻게 구성할지 개념을 잡을 필요가 있었다.

각 노드를 생성한 후에 루트부터 자식 노드까지 설정하고 최종적으로 루트 노드의 자식 노드 리스트를 맵 자료구조에 넣는 것으로 마무리하는 구조로 작성하였다.

추후 툴에서 생성된 노드 정보를 가지고 유동적으로 생성될 수 있게 변경할 것이다.

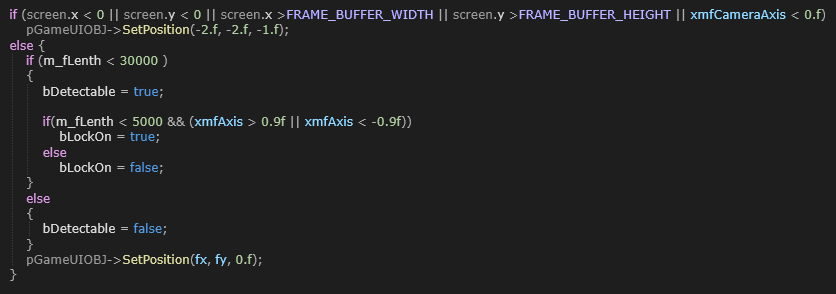
**김령운:**

1. **조준한 타겟에 카메라를 고정시키고 타겟이 플레이어보다 뒤로 갔을 때 타겟 지정이 유지되도록 수정, 락 온 범위 밖에서 조준할 타겟을 플레이어가 선택했을 때 UI가 깜빡이는 애니메이션 추가**

**그림1. 카메라 고정 시 타겟 지정이 유지되도록 수정**

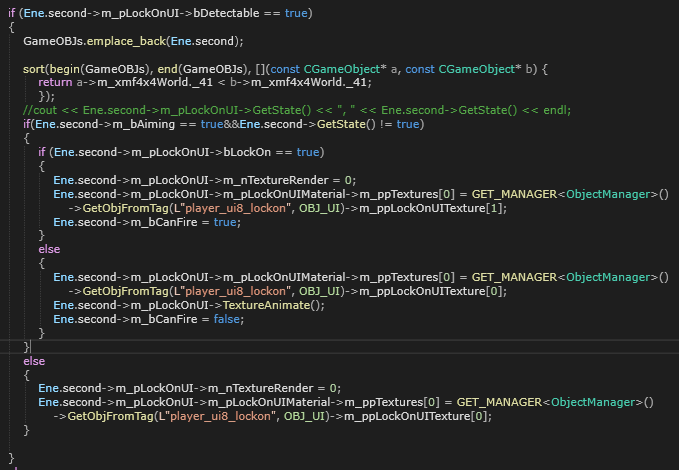
****

이전의 CLockOnUI 클래스의 MoveLockOnUI 함수는 플레이어의 Look 벡터와 적 오브젝트에서 플레이어로 향하는 방향 벡터를 내적 하여 두 객체 간의 사이각을 구하고 이 사이각이 둔각이 될 경우 타겟 지정이 풀리도록 함수를 작성하였는데 타겟으로 지정한 적 오브젝트에 카메라를 고정하는 기능이 추가돼서 적 오브젝트와 플레이어의 사이각에 상관없이 카메라 시야 안에서 조준 UI가 활성화되도록 수정하기 위해 카메라의 Look 벡터와 내적 하도록 수정했다.

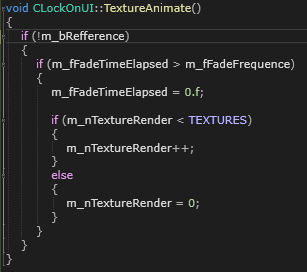
****

이전의 CLockOnUI 클래스의 MoveLockOnUI 함수를 수정하였다. bool bDetectable 변수를 추가하여 탐지 거리 내에 있는지를 첫번째로 검사하고 두번째로 플레이어와 5000 거리 내에 있고 플레이어가 객체를 바라보고 있는지를 검사해서 bLockOn 값을 설정했다.

**그림2. 탐지 거리 안에 있는 객체들에 대한 처리**



미사일을 직접 발사할 수 있는 조준 상태를 판단하는 bool bLockOnFire 변수를 bAiming 으로 수정하여 타겟 지정을 플레이어가 먼 거리에서 미리 할 수 있도록 했고 타겟 지정이 된 적은 초록색 조준 UI가 깜빡이면서 어떤 객체를 지정하고 있는지 표시해준다. 깜빡이는 애니메이션은 (**그림3.)** **CLockOnUI 클래스에 TextureAnimate() 함수**를 추가하여 구현했다.

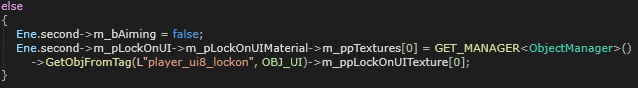


m\_nTextureRender 값이 1일 때만 조준 UI를 그려서 깜빡이는 효과를 구현했다.

bAiming 의 값이 false이고 소멸된 객체가 아니라면 깜빡이지 않는 초록색 조준 UI를 띄웠다.

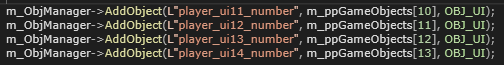
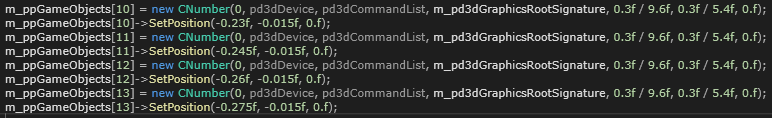
bAiming 의 값이 true(초록색 조준 UI가 깜빡이는 상태)인 객체는 탐지 거리 내에 타겟 지정이 된 상태이고 미사일 발사 범위 내에 이 객체가 있다면 깜빡이지 않는 빨간색 UI로 바뀌도록 작성했다. 최종적으로 빨간색 조준 UI가 뜨는 객체는 bCanFire bool 값이 true가 되고 이 값이 true인 객체에게 미사일이 유도되도록 수정했다.

**그림4. 탐지 거리 밖에 있는 객체들에 대한 처리**



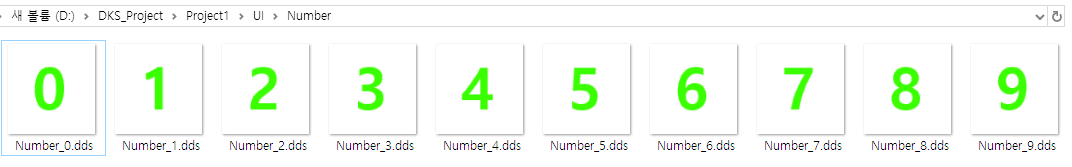
1. **비행기 속도를 HUD에 보여주는 기능 추가**

**그림5. 테스트씬 클래스의 빌드 씬 함수 부분**

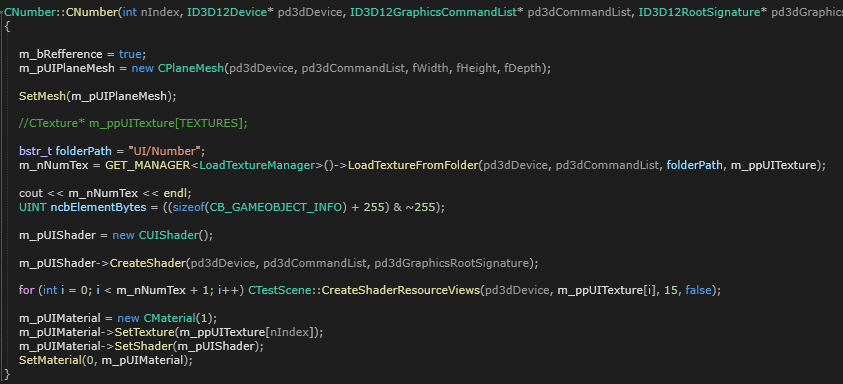
****

현재 전투기의 최대 속도 자릿수에 맞춰 자릿수만큼 숫자 텍스처를 빌드 했다.

**그림6. 숫자 텍스처 dds 파일**

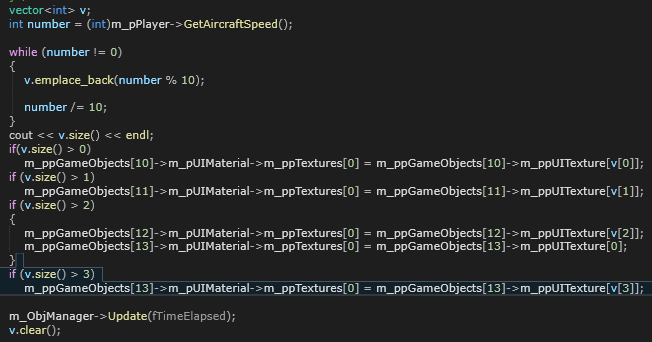
****

**그림7. CNumber 클래스 생성자**

****

이재원 팀원이 작성한 LoadTextureManager 클래스의 LoadTextureFromFolder 함수를 이용하여 0부터 9까지 되는 텍스처들을 생성했다.

**그림8. 테스트씬 클래스의 Animate 함수 부분**

****

**플레이어의 속도를 받아와 int 자료형으로 형 변환 후 number 지역 변수에 저장하고 나머지 연산을 활용하여 플레이어 속도의 1의 자리수가 vector의 첫번째에 저장되고 10자리수가 두번째 100자리수가 세번째에 저장되도록 하였다. 자릿수 별로 숫자를 벡터에 저장하여 0~9까지의 텍스처에 대입되도록 해서 화면에 플레이어 속도를 출력하는 효과를 구현했다.**

**현재 테스트씬 클래스의 애니메이트 함수에 직접 작성하여 구현했지만 따로 클래스와 함수로 작성하여 유동적으로 쓰일 수 있도록 수정할 예정이다.**

|  |  |  |  |
| --- | --- | --- | --- |
| **문제점 정리** | 이재원 :  - 아직 행동 트리 툴이 개발되지 않았으며 행위 또한 정의되지 않음  김령운 :  - 가끔 락 온 타겟을 바꾸지 않아도 자동으로 바뀌는 현상이 있음  -숫자 애니메이션 적용이 한정적임(유동적으로 다른UI에 적용 불가능) | | |
| **해결방안** | 이재원 :  - 아무 행위나 만들어서 트리를 테스트하고 완성된 양식에 맞춰서 노드를 정의하여 저장하는 행동 트리 툴을 제작한다.  김령운:  -적 오브젝트를 vector 자료구조에 저장하고 삭제할 때 문제가 있는지 확인한다.  -모든 UI에 숫자 애니메이션이 작동하도록 하는 클래스와 함수를 작성한다. | | |
| **다음주차** | 19 주차 | **다음기간** | 2020.05.04 ~ 2020.05.10 |
| **다음주 할일** | 이재원 : Behavior 툴 제작, 기총발사 제작  김령운 : Play Scene에서 Pause Scene으로 전환, 락 온 문제점 해결, Time/Score 표시, Ammo(잔탄 확인용) 숫자 UI 애니메이션 마무리 | | |
| **지도 교수**  **Comment** |  | | |