Helicopter Game

B677011 방찬웅

게임 구현 순서 및 구현 내용 (자세한 구현 방법은 주석으로 달아 놓았습니다.)

1. Helicopter Control  
   WASD 로 헬리콥터 이동, 맵 경계선 밖을 나가지 못하게 입력 예외 처리  
   맵이 자동 생성되어 왼쪽으로 움직이기 때문에, 왼쪽 이동키를 눌렀을 땐 속도가 증가되어 이동
2. Helicopter Animation  
   image를 만든 뒤 Vector Container 안에 넣고 한 개씩 순회하여 이미지를 출력하여 애니메이션 구현
3. Helicopter Shooting  
   Space 키를 누르면 fire 트리거가 활성화 되어 Bullet GameObject를 새로 생성하여, bullets Vector Container에 저장하여 연사 기능 추가.
4. Bullet   
   오른쪽으로 계속 움직임, 맵 경계선 밖으로 나가지 못하게 예외 처리  
   사용이 끝난 총알은 bullets Vector Container에서 제거
5. Map Editor  
   좌클릭으로 맵에 terrain을 세팅할 수 있고, 우클릭으로 지우기 가능  
   버튼으로 원하는 terrain을 선택할 수 있으며, Clear, Save, Load 기능 제공
6. Map Generator  
   여러 개의 수직 선이 모여 하나의 맵을 만드는 방식  
   pivot(중심)과 space(빈 공간)을 설정하여 pivot에서 space 만큼의 빈 공간을 위 아래로 생성하며, 그 외는 terrain으로 채움. Terrain은 총 3가지가 있으며, 바깥쪽은 진하게 안쪽일수록 연한 terrain으로 채움.  
   수직선은 이전 수직선의 정보를 받아 자신의 정보를 세팅하여 끊김없이 자연스러운 맵 생성.  
   수직선은 입력한 개수만큼 생성되며, 계속해서 순환되어 재사용 됨 (메모리 단편화 현상 방지)
7. Enemy  
   적은 20 frame에 한번 씩 **출현하며**, map의 pivot값에 따라 위아래 움직임  
   helicopter에 연사 기능이 있으므로 Enemy가 자주 생성 되도록 난이도 조절  
   20 frame 마다 총알 발사, 플레이어 총알에 맞으면 없어짐  
   총알 발사 시, 포신이 줄어들었다가 다시 되돌아오는 애니메이션 추가
8. 충돌 처리  
   1) 헬리콥터 – 맵  
   헬리콥터 shape의 채워져 있는 칸의 Position을 모두 저장한 Vector Container를 만듦.  
   맵의 수직선에 빈 공간 바로 위와 아래 채워져 있는 부분의 Position을 가져옴.  
   헬리콥터의 채워져 있는 부분과 수직선의 경계 부분의 Position이 서로 같다면 충돌 처리  
   2) 총알 – 맵  
   각각의 총알의 Position과 수직선 경계 부분의 Position이 같다면 충돌, 충돌 후에 총알은 사라짐  
   총알은 오른쪽, 맵은 왼쪽으로 서로 이동하고 있으므로, 뚫고 지나가는 버그가 발생하여 다중으로 충돌 체크  
   3) 헬리콥터 - 총알  
   헬리콥터 shape의 채워져 있는 칸의 position이 총알과 같아진다면 충돌 처리   
   4) 적 – 총알  
   적 shape의 채워져 있는 칸의 position이 총알과 같아진다면 충돌 처리  
   적의 HP가 0이 되면 없어지고, 일정 시간 뒤에 다시 출현
9. UI  
   1) time UI   
   <ctime>을 추가해 clock() 함수로 start와 end를 빼서 시간을 구하는 방법과  
   <chrono>를 추가해 시간을 측정하는 방법을 시도 해봤으나, time\_t start, chrono::time\_point<chrono::system\_clock> start 를 Scene내에서 선언할 시 다른 것들이 정의되지 않는 오류가 발생하여 다른 방법시도  
   텍스트이(가) 표시된 사진

   자동 생성된 설명  
   현재 게임 로직 상 Sleep() 함수를 사용하고 있으므로 위의 방법을 사용한다 하더라도 정확한 시간 측정이 어려울 것으로 판단되어, Sleep() 함수의 특성을 이용하여 매 프레임을 0.1초로 보고 프레임마다 시간을 측정  
   2) HPBar   
   원하는 크기의 HPBar를 원하는 위치에 세팅하고 HP가 감소하는 기능과, reset 할 수 있는 기능 구현  
   3)Game Over  
   게임 오버 시 게임이 종료되었음을 알려주는 게임오버 UI 표시

구현 시 어려웠던 점 및 특이사항

1. 컴포넌트 설계의 진행 로직에 관한 전반적인 이해   
   이해를 하는 시간이 오래 소요되었지만, 유니티와 연관을 지어 이해하려고 노력하다 보니 이해한 뒤에는 사용이 쉬워졌다.
2. 최대한 상수가 아닌 변수들로 게임이 진행될 수 있도록 설계   
   GameObject를 선언하고 선언한 정보를 바탕으로 다른 부가적인 요소들도 초기화되고, 값이 고정되는 것이 아닌 하나의 값만 바꾸면 연관된 로직 전체가 바뀔 수 있도록 설계하였다.
3. 메모리  
   동적 생성하는 것을 최소화하여 메모리를 과다하게 사용하지 않도록 신경을 쓰고, 사용하지 않는 메모리는 최대한 반환하여 사용량을 줄였다.
4. 충돌 처리  
   제일 많이 고민을 한 충돌 처리는 처음에 테트리스에서 했던 것처럼 pos + 1 이 빈칸이 아니라면 충돌하는 방법으로 생각을 했으나, 그렇게 되면 위, 아래, 뒤, 충돌 처리 시 부딪히지도 않았는데 충돌 처리가 되기 때문에 포기 하였고, map을 bool\* 배열로 만들어 0, 1로 충돌 처리를 하려고도 생각했지만 이전 방법과 별로 다른 점이 없고 맵을 하나 더 생성하는 번거로움이 있기 때문에 포기 하였다. 그 다음 생각해본 방법은 Position을 비교하여 맵과 Object의 채워져 있는 부분의 Position이 같아지면 충돌 처리를 하는 방법으로 구현했다.

학기를 마치며

복학해서 무지한 상태로 학기를 시작하여, 아직 많이 부족하지만 많이 성장한 것 같아 뿌듯합니다.  
 처음 교수님께 상담을 받고 1학기 수업 영상들을 받아 어떻게든 따라가자 란 마음으로 1주일 만에 모든 1학기 수업 내용을 공부하고, 노력했던 것이 도움이 많이 되었던 것 같습니다. 지뢰찾기, 테트리스, 헬리콥터 게임들을 만들면서 무에서 유를 만들려고 하루 종일 고민하고, 생각을 그려보고, 직접 구현해보고, 디버깅하는 과정들을 겪으면서 많이 성장하고 배웠습니다. 수강 신청을 할 때는 두려웠지만 이 수업을 듣길 굉장히 잘했다는 생각이 듭니다.   
한 학기 동안 정말 고생 많으셨습니다. 감사합니다.