**객체지향 프로그래밍(2) 최종 기말 과제 구현 보고서**

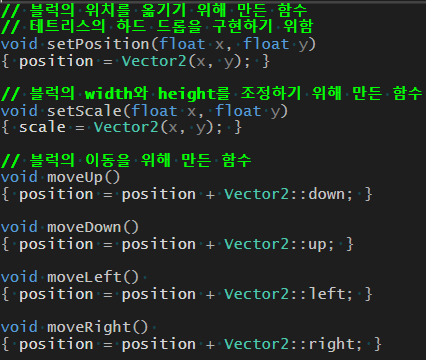
작성자: 김홍준

날짜: 2019.12.11

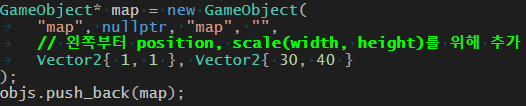
구현한 코드를 설명하기 앞서 공지사항의 [객체지향프로그래밍(2)] 컴포넌트 기반 설계를 활용한 MonsterWorld 제작본의 첨부파일을 기반으로 작성했으며 테트리스 코드는 서범주 교수님의 상속 모델로 구현한 테트리스 코드(이하 테트리스 코드)를 참고했음을 밝힙니다.

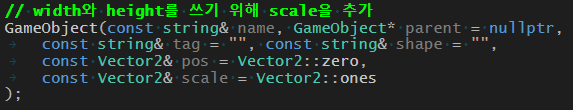
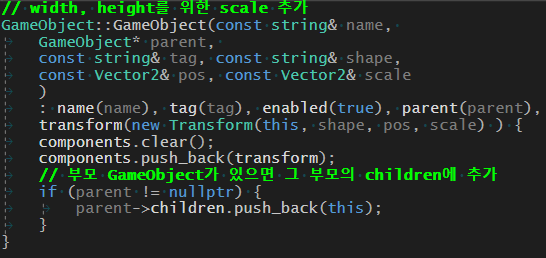
1. 수행해야할 과제  
   - 컴포넌트 기반 설계를 이용해 테트리스 만들기
2. 기능 세분화
   1. 테트리스 블록들과 맵을 만들고 출력하기
   2. 테트리스 블록의 이동 및 회전
   3. 테트리스 블록 판정, 다음 블록, 다음 패널 추가
   4. 테트리스 블록 줄 지우기, 점수, 최고 점수, 게임 오버 추가

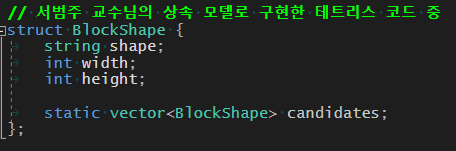
2 - ① 테트리스 블록들과 맵을 만들고 출력하기 구현

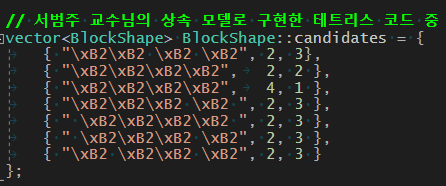
먼저 테트리스 블록과 맵의 width, height, position을 컨트롤 하기 위해 Transform.h 파일에 여러 함수를 추가했다. 여기서 width와 height는 scale로 관리하기 위해 setScale함수를 만들었다.  


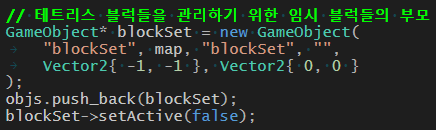
그리고 테트리스 코드에서는 테트리스 블록을 회전하는 함수가 있어서 rotation도 컨트롤 하기 위해 함수를 추가했다.  

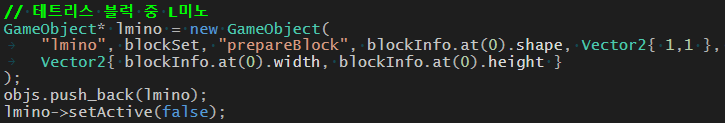
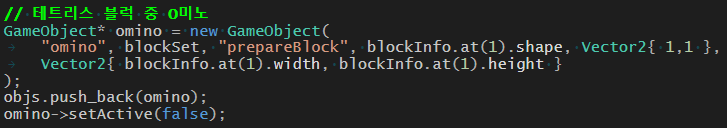
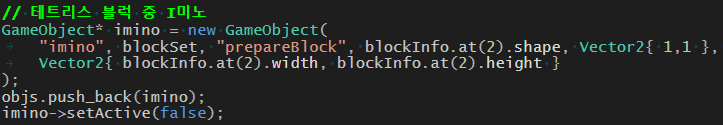
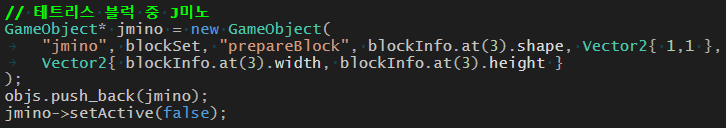
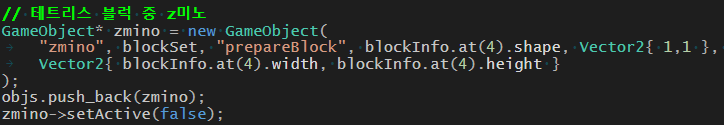
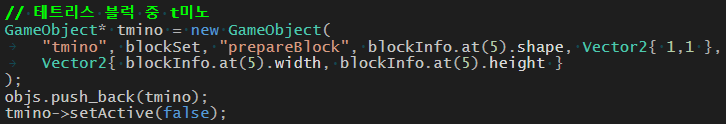
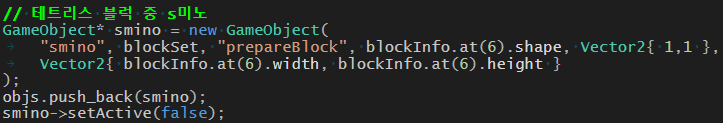

위 기능을 추가시킨 후 일단 GameEngine.cpp에서 맵을 만들고 블록을 출력하는 창을 만들기 위해 GameObject map을 만들었다. 여기서 position은 맵의 출력 위치를 정하고 scale은 width와 height를 담당하므로 맵의 크기를 정한다.  


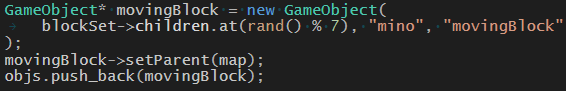
위 코드가 잘 수행하게 만들기 위해 GameObject생성자에서 Scale을 추가로 받게 했고 parent가 있을 경우 자기 객체를 부모의 children에 들어가도록 만들었다.  
  


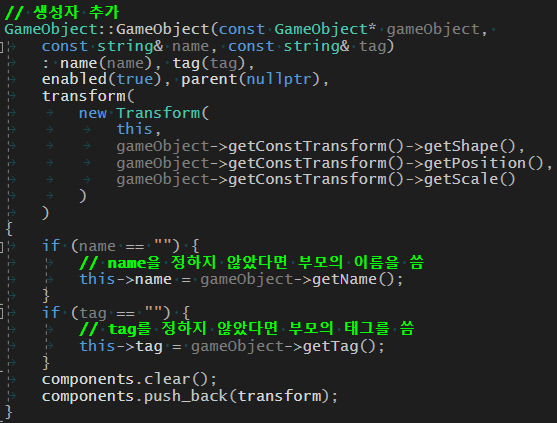
맵을 만든 후 테트리스의 블록과 모양을 구현하기 위해 상속 모델로 구현한 테트리스 코드 내에서 BlockShape구조체를 꺼내왔다. 이 코드는 Utils.h파일에 저장하였다.  


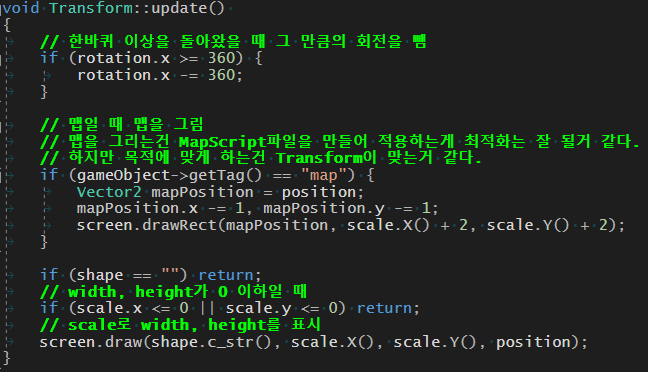
Candidates 내용은 테트리스 코드 내 candidates를 Utils.cpp파일에 정의했다.  


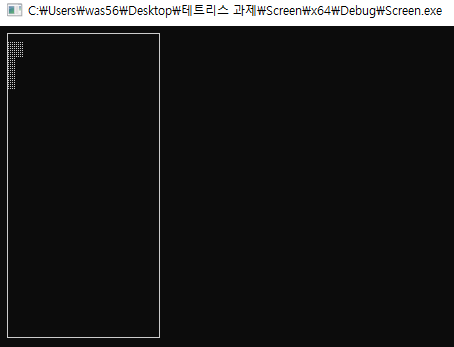
이어서 GameEngine.cpp에 블록들을 만들고 관리하기 위해 BlockSet을 만들었다. 그 이유는 BlockSet을 Block들의 부모로 지정하면 관리하기가 편하기 때문이며 그 이외의 기능은 없기 때문에 Active는 false로 두었다.  


테트리스 블록들은 테트리스 코드와 scale을 이용해 BlockSet의 자식으로 들어가며 테트리스 게임이 돌아갈 때는 새로운 GameObject를 만들어 아래 블록들 중 한 개를 복사해 랜덤으로 나오게끔 만들 예정이다. 따라서 Active는 false로 두었다.  
  
  
  
  
  
  
  


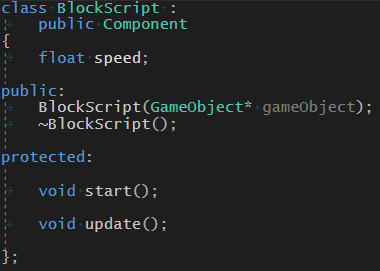
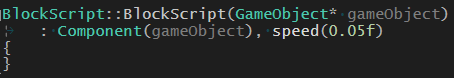
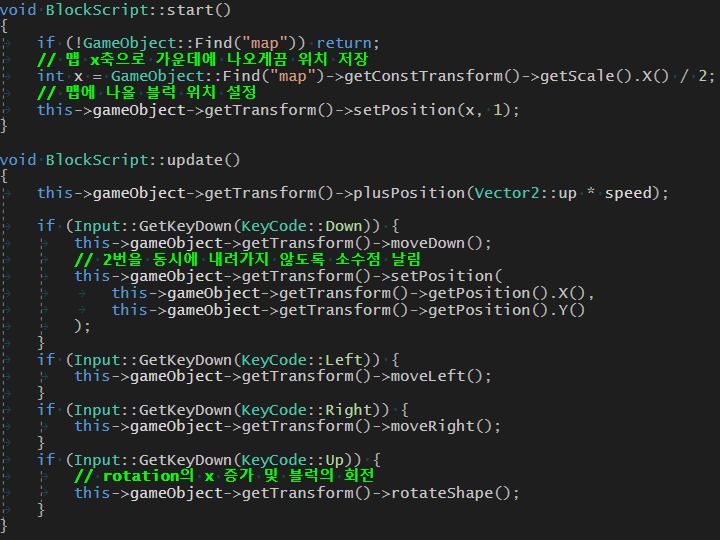
위 블록들 중 한 개의 블록만 복사하여 저장하도록 하기 위해 GameObject 객체를 만들었으며 맵에 띄울 것이기 때문에 부모를 map으로 설정했다.  


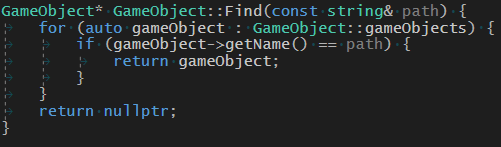
위 코드가 잘 작동시키기 위해 GameObject에 생성자를 추가하였다.   


이로써 2 - ①을 구현하기 위한 객체들을 모두 생성했고 Transform.cpp의 update를 수정시켜 맵과 테트리스 블록을 출력하도록 만들었다.  


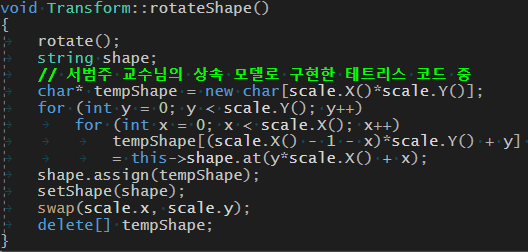
위 코드들을 적용하여 컴파일한 결과 아래 사진과 같이 잘 작동하는 것을 볼 수 있다..  
  
2 - ① 테트리스 블록들과 맵을 만들고 출력하기 구현 완료

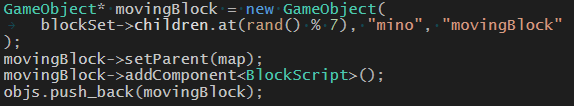
2 - ② 테트리스 블록의 이동 및 회전 구현

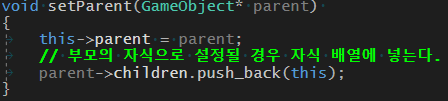
먼저 블록에 적용할 스크립트를 만들며 start함수에는 블록의 처음 위치를 설정하고 update함수에는 키를 받고 움직이거나 회전하는 함수를 실행시키도록 만든다.  
   
 

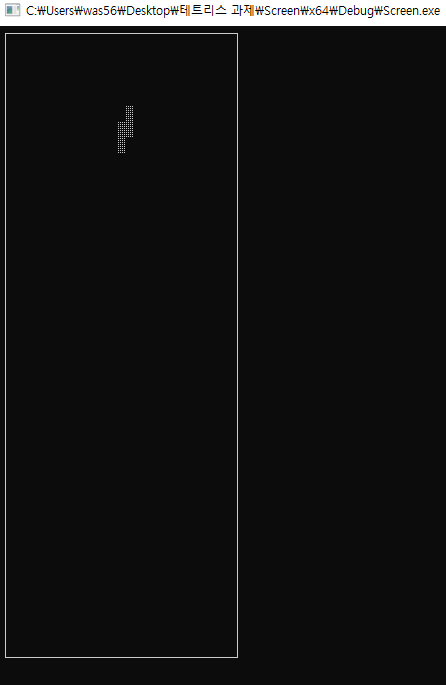
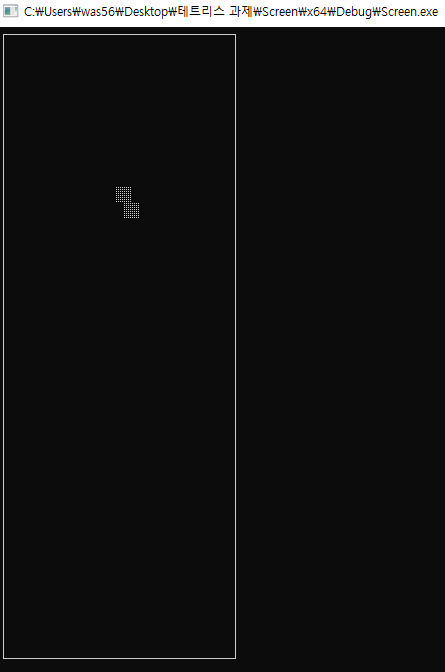
위 코드가 잘 작동할 수 있도록 GmaeObject파일에 Find함수를 구현한다.   


그리고 Utils.h에 Vector2 operator\*을 구현한다.  


마지막으로 Transform 파일에 rotateShape함수를 구현한다.   
  


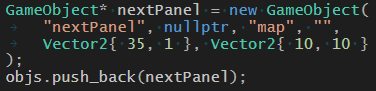
이후 GameEngine.cpp 파일에 추가했던 movingBlock객체에 BlockScript 컴포넌트를 추가시킨다.  


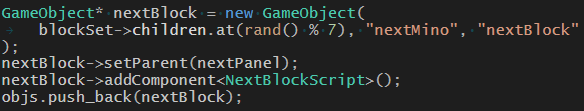
참고로 setParent함수에 부모의 자식 배열에 자식이 들어가도록 수정하였다.  


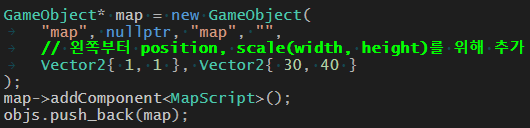
위 코드들을 적용한 컴파일한 결과 아래 사진들을 통해 잘 작동하는 것을 볼 수 있다.  
 

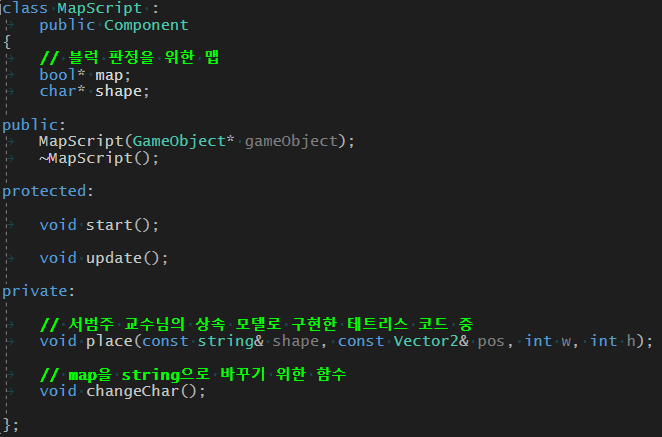
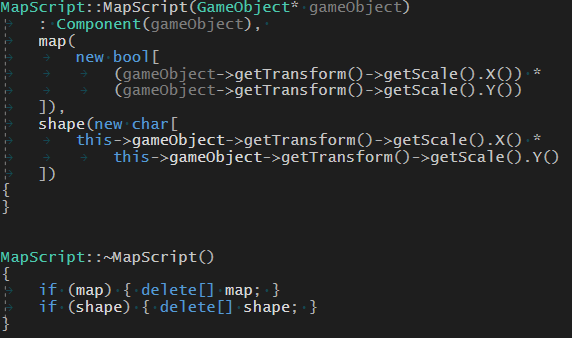
2 - ② 테트리스 블록의 이동 및 회전 구현 완료

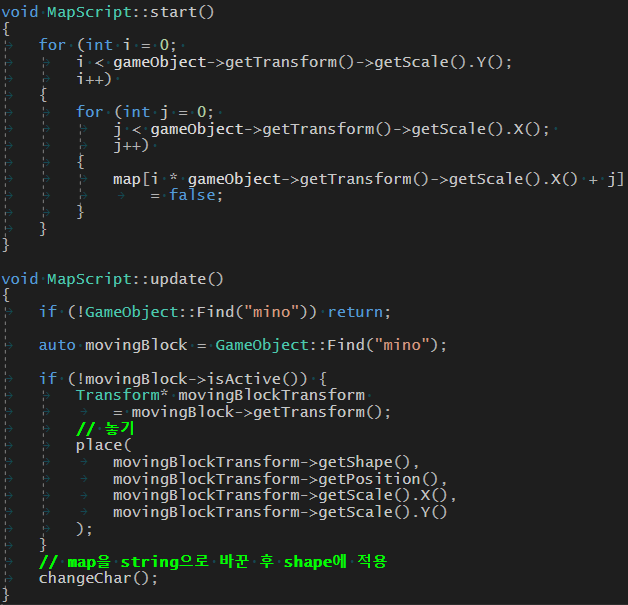
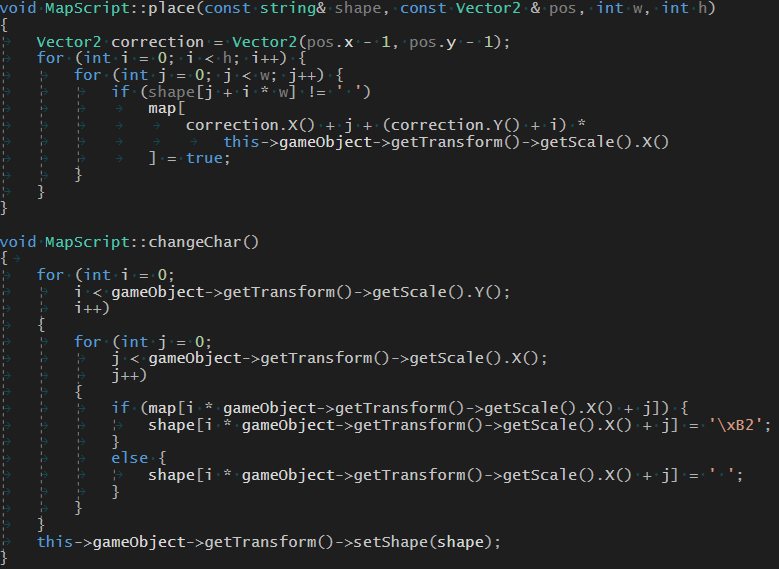
2 - ③ 테트리스 블록 판정, 다음 블록, 다음 패널 추가 구현

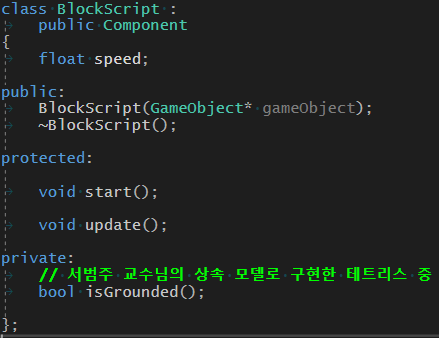
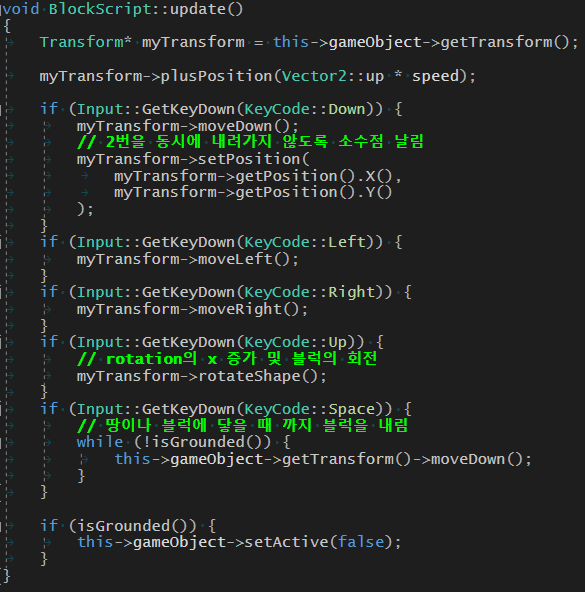
먼저 다음 패널을 만들고 맵처럼 출력하기 위해 태그를 map으로 설정했다.  


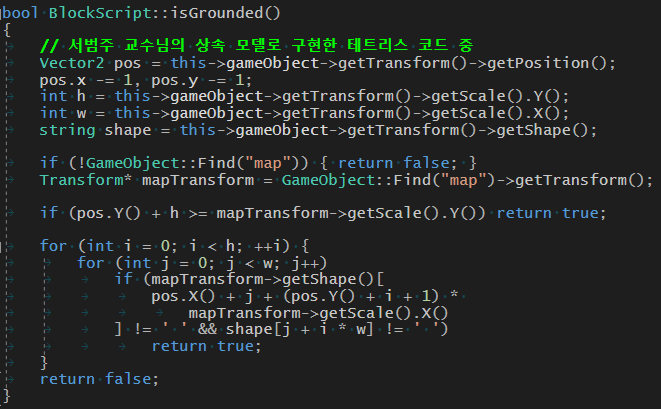
다음 블록 표시 및 정보를 알기 위해 생성하였다.  


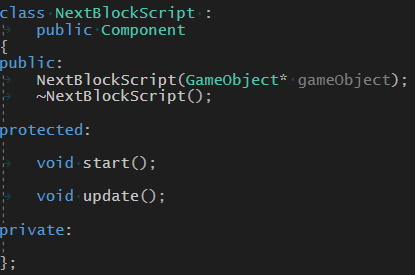
NextBlockScript스크립트를 작성하기 전에 테트리스 블록이 바닥이나 블록에 의해 드랍 판정이 되게끔 하기 위해 map에 MapScript컴포넌트를 넣었다.  


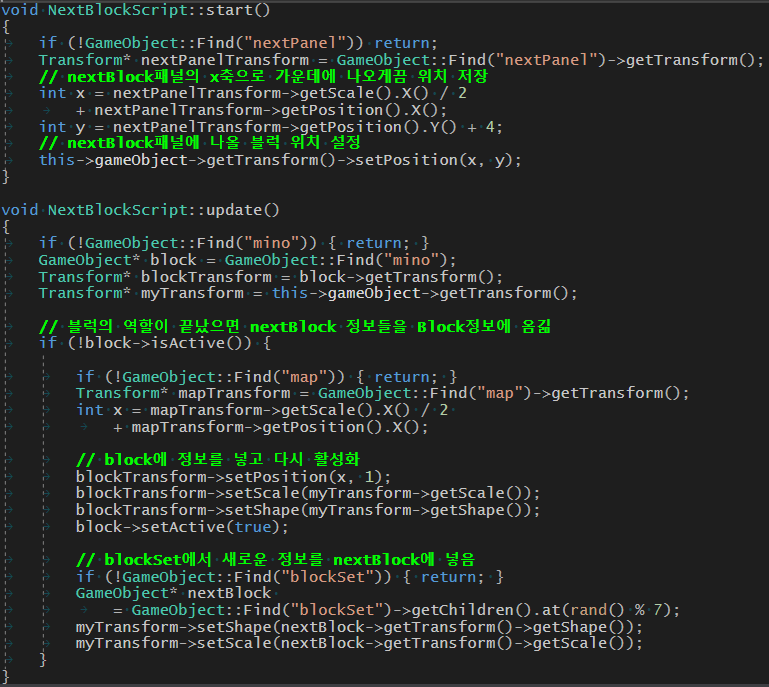
그 후 MapScript 스크립트는 블록이 바닥에 닿고 그 블록이 어디에 놓고 어떤 모양이 되었는지 기억해야하기 때문에 map이라는 변수를 만들고 동적할당하여 map의 역할을 하도록 만들었다. place라는 함수를 이용해 블록이 놓였을 때 map에 블록이 저장된다. changeChar는 변수 map을 기반으로 스크린에 출력해야하기 때문에 map을 출력하도록 string으로 바꾸고 shape에 적용하는 함수이다.  
  


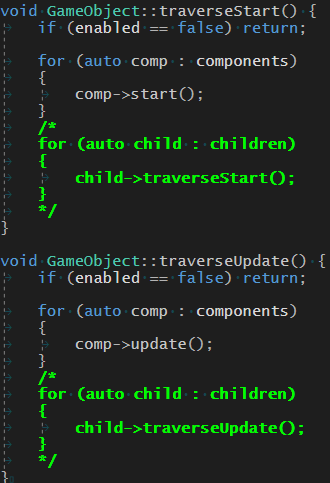
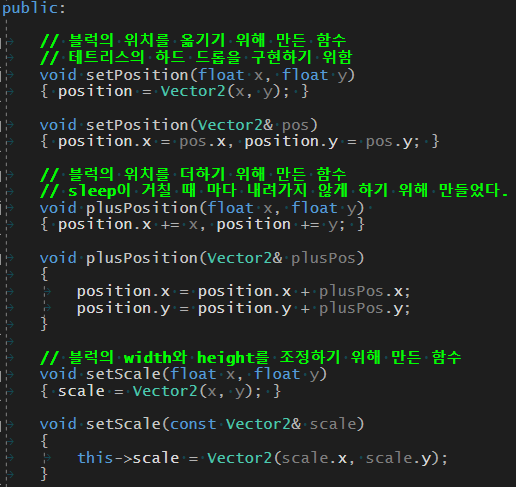
  


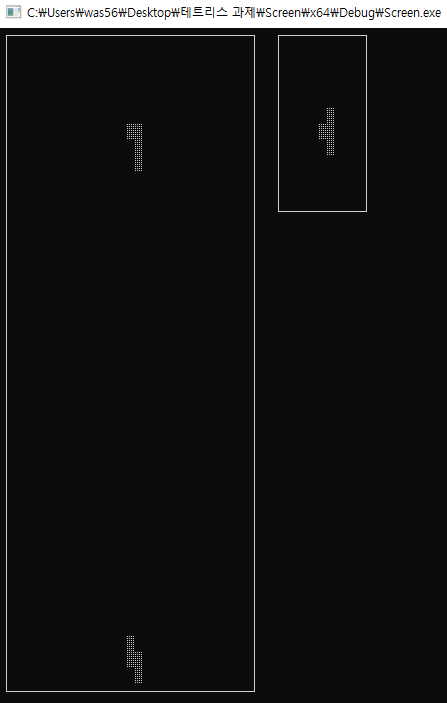
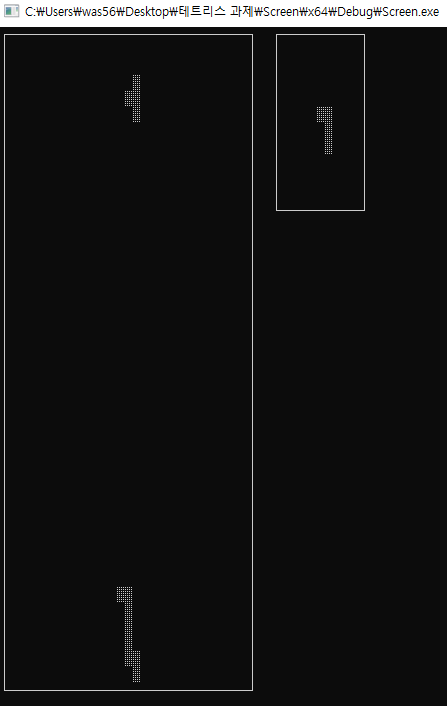
이후 블록이 바닥이나 블록에 의해 드랍 판정하는 부분을 BlockScript에 구현하였으며 테트리스 코드에 있는 함수를 참고했다.  
  




이 부분에서 끝내면 블록이 드랍 판정이 되고 새로운 블록이 나올 수 없으므로 NextBlock의 NextBlockScript컴포넌트를 이용하여 드랍 판정 된 블록을 처음 나왔던 것처럼 설정하도록 만든다.  


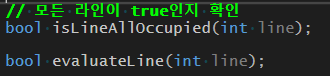
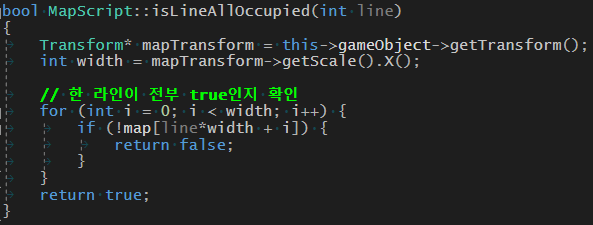
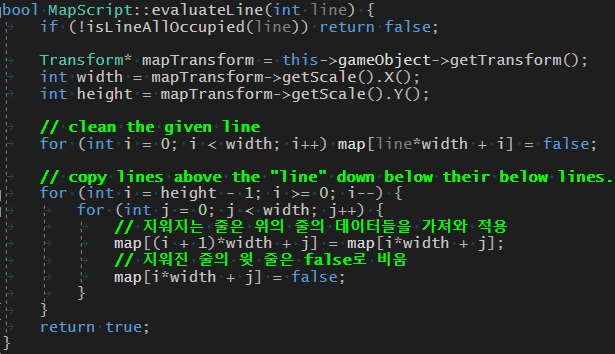


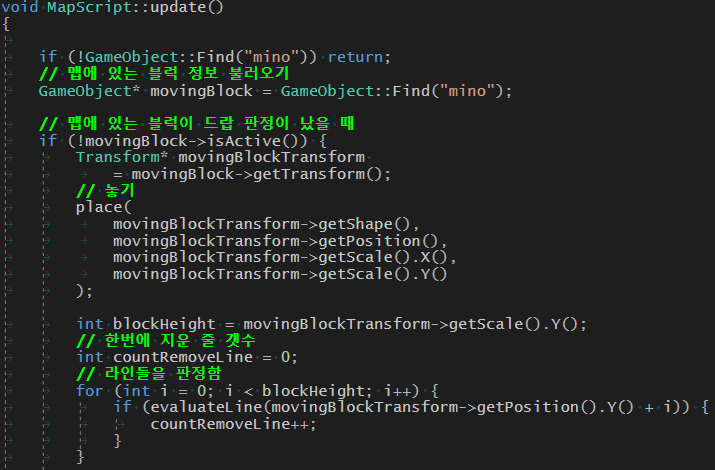
추가적으로 GameObject함수에서 버그가 일어나는 부분이 있어서 주석 처리하였고 위 코드들이 잘 작동하도록Transform파일에서 오버로딩을 이용해 함수를 구현하였다.  
 

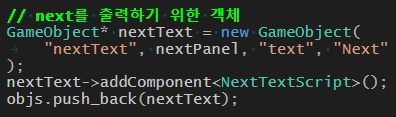
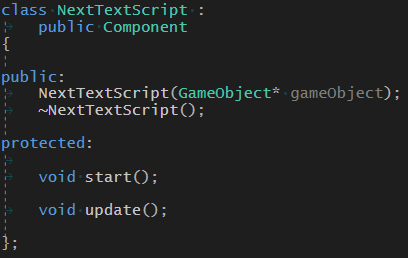
위 코드들을 적용시키고 컴파일한 결과 아래 사진들처럼 잘 작동하였다.  
 

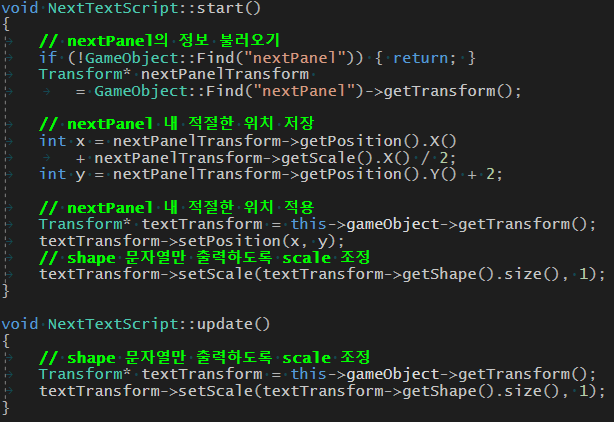
2 - ③ 테트리스 블록 판정, 다음 블록, 다음 패널 구현 완료

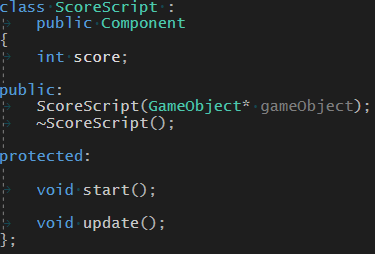
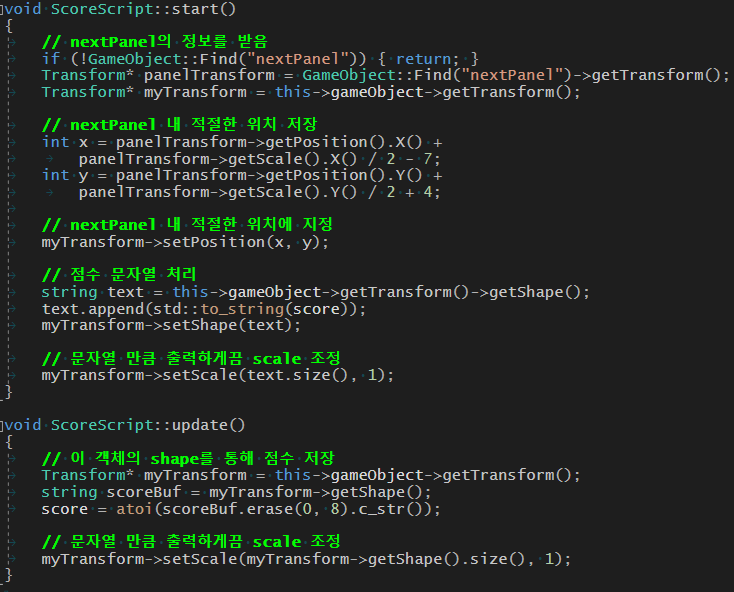
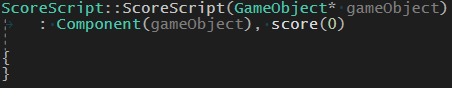
2 - ④ 테트리스 블록 줄 지우기, 점수, 최고 점수, 게임 오버 추가 구현

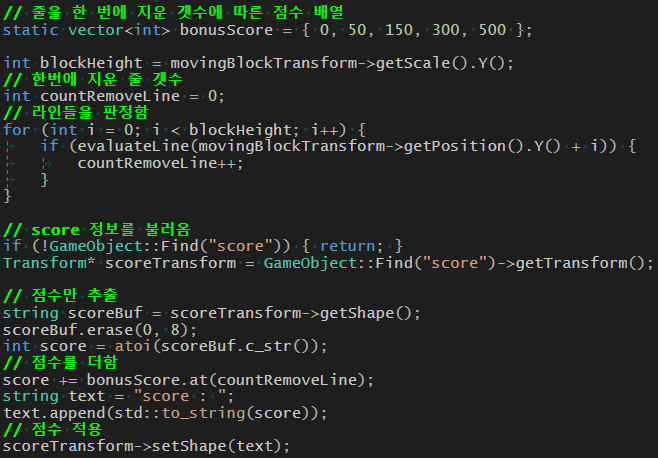
먼저 테트리스 블록 줄을 지우기 위해 테트리스 코드에서 한 라인이 채워지면 지우는 함수인 evaluateLine함수를 참고했으며 map을 읽어야하기 때문에 MapScript의 update함수를 이용해 줄을 지우도록 구현하였다.  
  
  


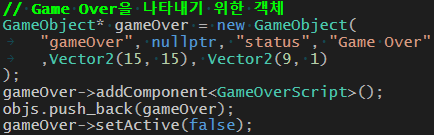
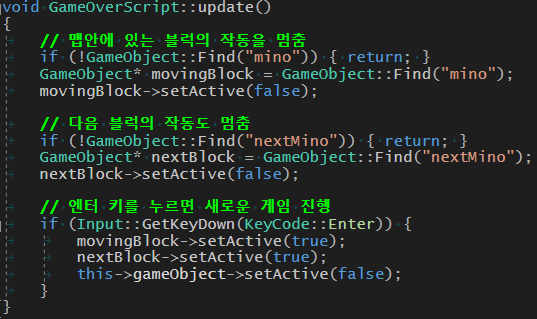


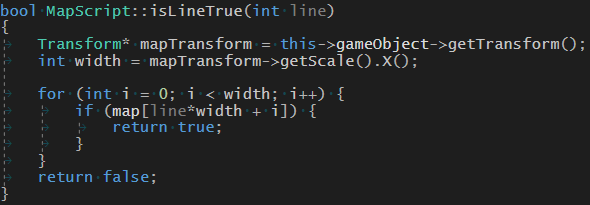
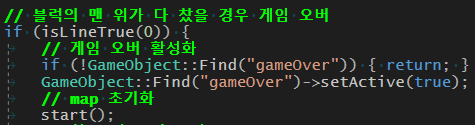
이 부분까지 완성을 하면 정상적으로 게임을 즐길 수 있을 정도로 구현이 된다. 그 후로는 추가적인 부분이며 먼저 블록에 next를 표시하게끔 하기 위해 nextText라는 GameObject와 NextTextScript스크립트에서 잘 표시하게끔 구현하였다.  
  


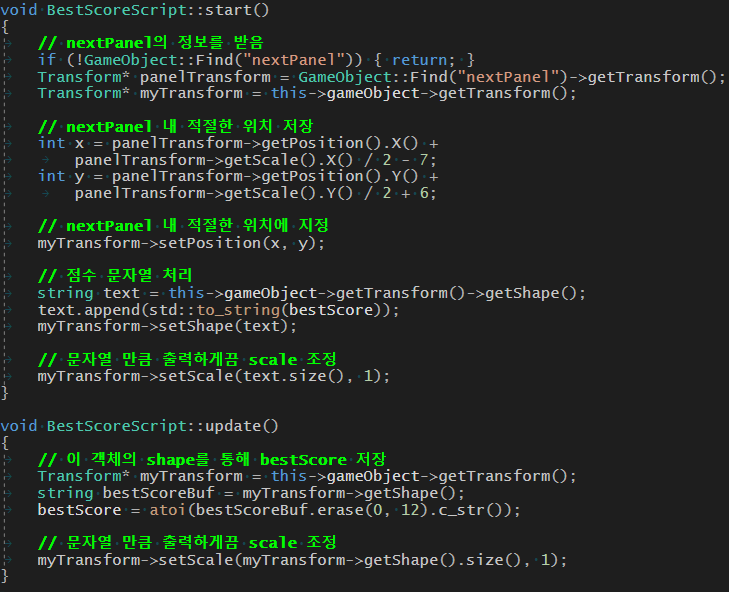
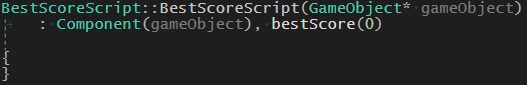
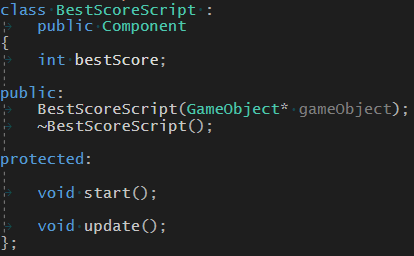


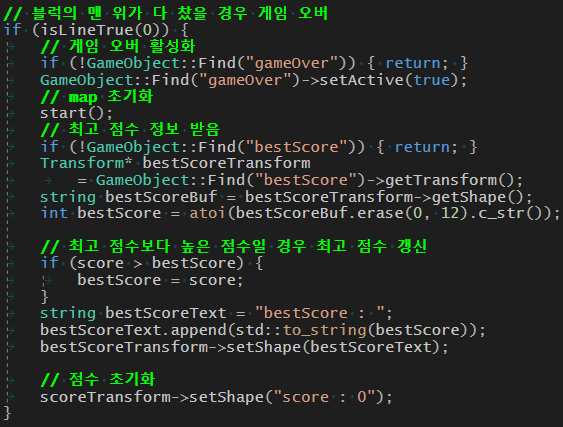
다음으로는 점수를 구현했으며 GameObject인 score과 ScoreScript를 통해 점수를 출력하고 점수를 저장하도록 구현했다.  
  


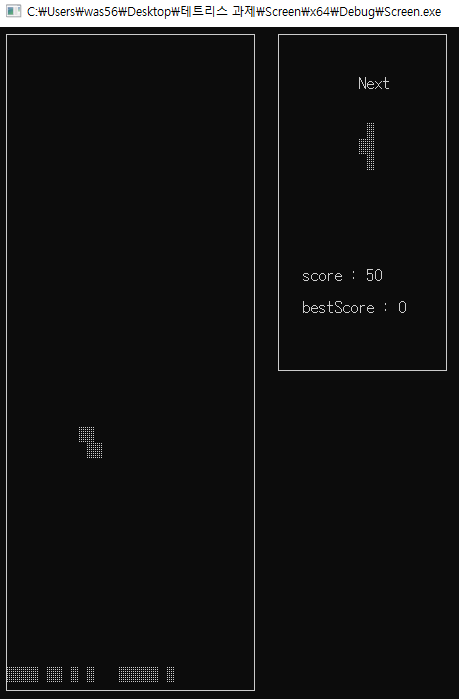
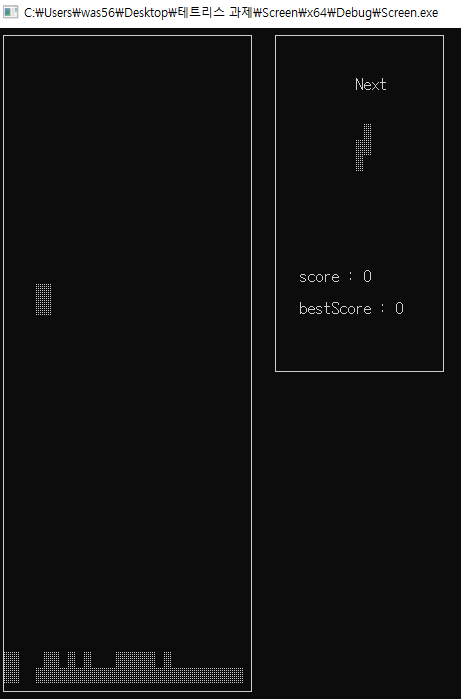
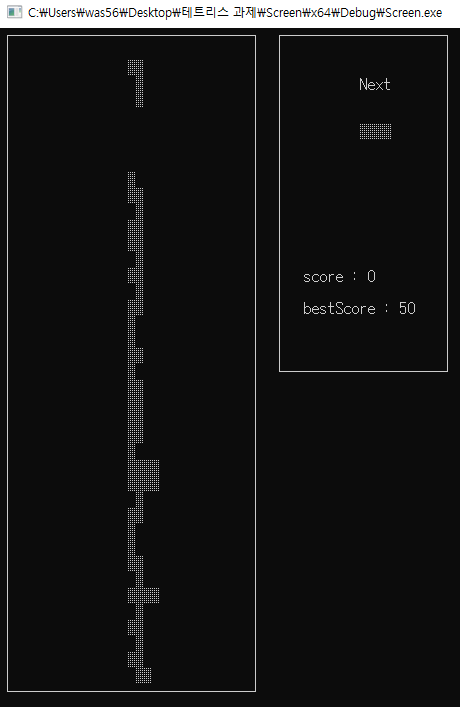
이 부분까지만 하면 점수를 갱신할 수 없기 때문에 테트리스 블록 줄이 없어지는 것을 판정하는 MapScript에 점수를 갱신하는 부분과 여러 줄을 한번에 지웠을 때 보너스 점수가 더하고 점수를 갱신하도록 구현하였다.  


그 다음으로는 게임 오버를 GameObject인 gameOver와 GameOverScript를 이용하여 게임 오버의 active를 true로 설정하면 모든 게임의 초기화와 동시에 게임이 멈추도록 만들고 Enter키를 누르면 다시 시작하도록 구현하였다.  
  


게임 오버 판정은 MapScript에 블록이 드랍 판정되었을 때 isLineTrue함수를 통해 그 블록이 맵의 맨 윗줄에 존재하면 게임이 끝나도록 구현했다.  
  


마지막으로 최고 점수를 구현했으며 GameObject인 bestScore와 BestScoreScript 스크립트를 이용해 최고 점수를 출력하고 저장하도록 구현했다. 참고로 BestScoreScript 스크립트는 ScoreScript 스크립트와 매우 유사하게 구현했다.  


최고 점수도 마찬가지로 이 부분까지 구현하면 최고 점수를 갱신할 수 없기 때문에 게임 오버를 판정하는 MapScript에 게임이 끝나는 지점에서 최고 점수를 갱신하도록 구현하였다.  


이 부분까지 구현하고 컴파일한 결과 아래 사진들과 같이 잘 작동하는 것을 볼 수 있다.  
  


테트리스 플레이 영상은 아래 사이트에 업로드해 놓았다.

<https://was564.github.io/study_html/Tetris.html>

참고 사이트:

<https://github.com/beomjoo90/OOP/commit/3ccf371b4b9bffed226b781ab17029090580c23b>