<구현 보고서>

B993185 유민경

*구현 링크: https://github.com/Yuminkyeong/2022_GPP.git

//영상 및 asset 사용으로 인해 너무 느리게 업로드 되어 전체 코드는

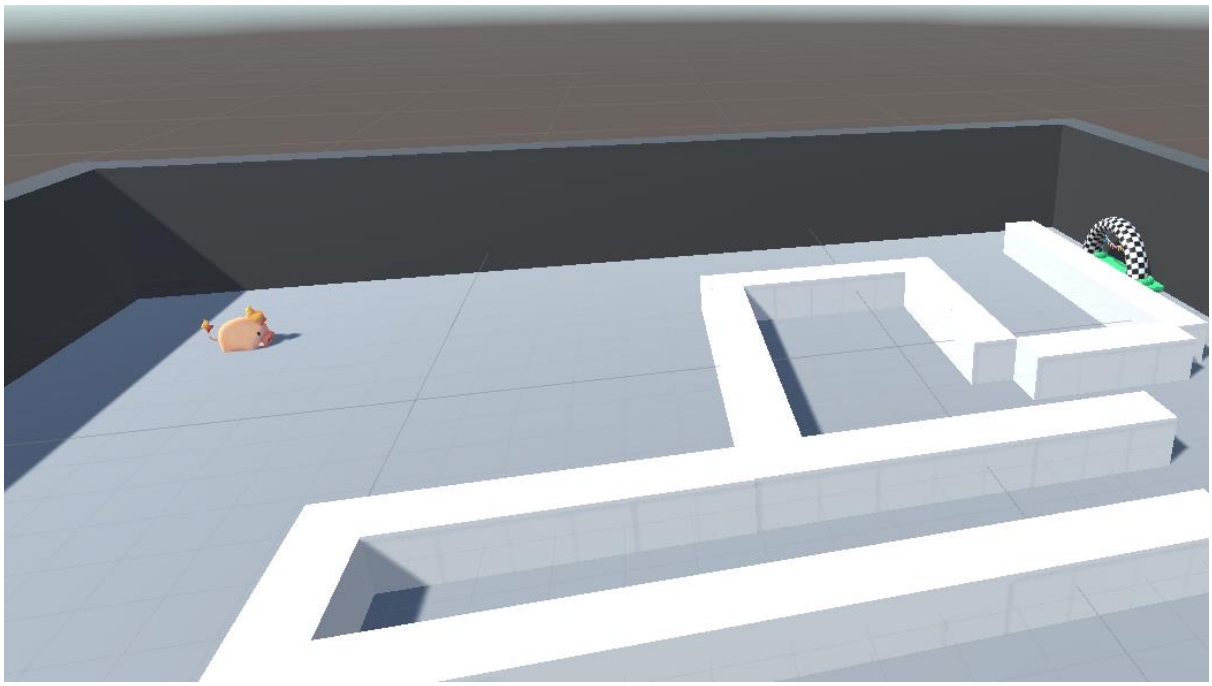
github 링크에 올려 공유하게 되었음

branch에 refactoring 전/ 후 로 나누어 두어 확인할 수 있음

1. 구현 계획

1) 스토리

: '최소로 이쑤시개를 옮겨 강아지 만들기' 퍼즐에서 착안한 게임으로, 벽을 최소한으로 부시고 일반 벽을 최소한으로 생성해서 빠른 시간 안에 게임 목표 지점에 도달하는 게임을 만들고자 함.



2) 길찾기 알고리즘과 Command Pattern

: A* 알고리즘을 사용하고, 벽을 생성, 파괴, 플레이어의 이동 등과 같은 플레이어 중심의 입력은 Command Pattern을 활용해 작성.

PlayerScript.cs; Undo, Redo, 마우스 왼쪽으로 이동, 마우스 오른쪽으로 일반 벽 생성

CommandManager.cs; Command Pattern을 전체적으로 관리하는 Manager로 Add Command, Execute, Undo 기능들을 수행

CreateWallCommand.cs; 일반 벽을 생성하는 Command로 마우스 오른쪽으로 생성할 수 있고, 생성함에 따라 일반벽을 설치하여 골인 지점까지의 단거리를 설계할 수 있음

Wall2Command.cs; 부술 수 있는 벽으로 흰색->초록색->빨간색->파괴 순으로 변하며, 파괴 시 벽이 있던 지점을 넘어 이동할 수 있음

ReplayCommand.cs; 게임 성공 시 replay가 자동으로 재생되고, 게임이 종료됨

UndoCommand.cs; 게임 뒤로 가기 기능

3) Object Pool

: 자주 생성, 파괴가 이뤄지는 wall2 가 포함된 map으로 Object Pool을 이용하여 구현

4) Observer Pattern

: Player의 위치에 따라 동작되는 장애물로, player가 subject가 되어 동작시키는 것으로 Observer Pattern에 맞다고 생각하여 이처럼 구현

FallingBox.cs; 공중에 떨어지는 장애물이 생성 및 동작됨

5) State Pattern

; 일정 거리를 배회하는 Monster AI로 player와 가까워졌을 때 Player를 쫓아와서 빠른 play를 방해함

2. 구현 시 고민이 되었던 부분

1) Observer Pattern으로 떨어지는 장애물을 구현했을 시, Player의 움직임에 따라 달라지는 장애물이므로 이렇게 구현하고자 하였음. 이러한 관점에서 보았을 때, State Pattern 역시 Player의 위치에 따라 Object의 움직임이 달라지는 것인데 굳이 나눠서 Pattern을 적용하는 것이 의미 있을지 고민하게 됨.

2) Object Pool의 형태가 맵 형성에 알맞은 형태인지 고민하게 됨. 자주 생성/파괴 된다는 특성을 가지고 있으나, 개수가 제한되어 있는 리듬 게임의 노트나 장전 개수가 정해져 있는 총 게임의 경우가 더 알맞지 않을까 라는 생각이 들었음.

3) Command Pattern의 근본적인 문제를 고민하게 됨. Command 의 개수가 늘어날수록 script가 계속해서 늘어나야 하는데 이것이 좋은 방법인지 잘 모르겠다는 생각이 들었음. 하지만 Undo, Replay에 있어 좋은 방식임에는 틀림없어 이대로 적용하게 됨.

4) Event Queue의 추가 고민. 게임의 시작, 종료를 Event로 처리하고, 중앙 Event에서 scene을 로드 하는 것을 고민했으나, 굳이 Global하게 중앙 event에 불러 둬야 할 필요성을 느끼지 못해 그대로 진행했음

3. 구현 시 문제 되었던 부분

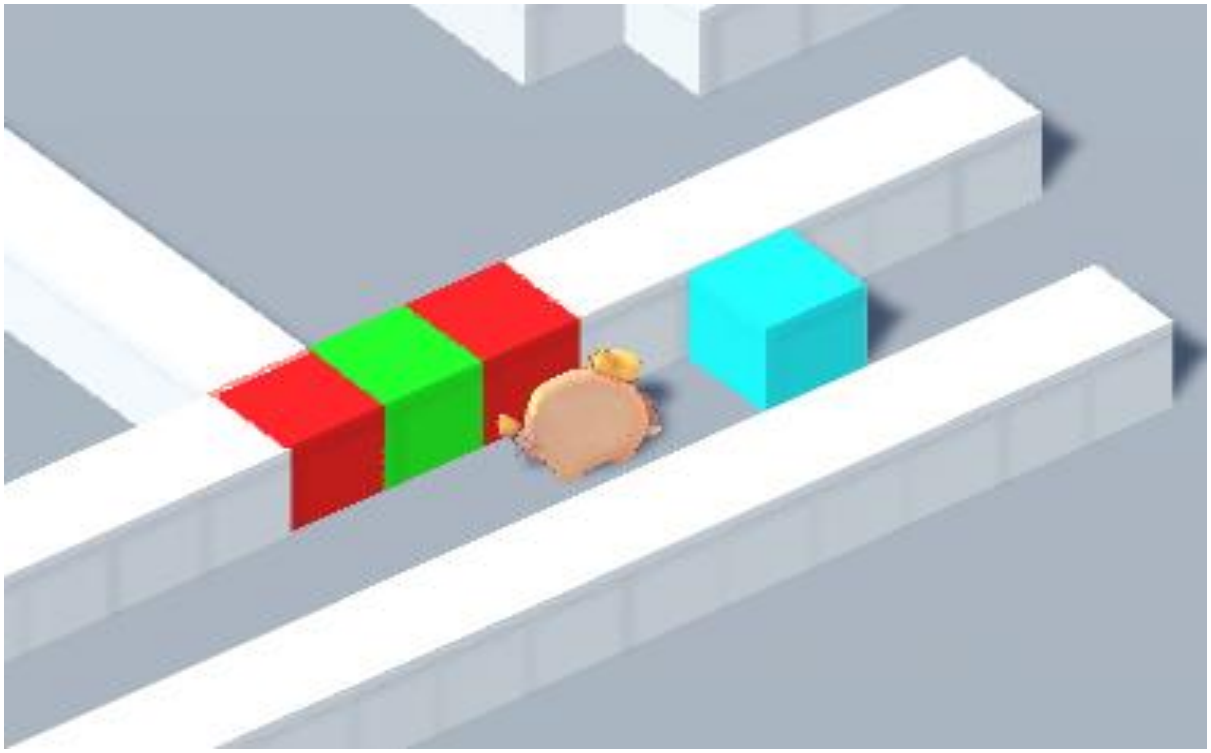
1) Wall2Command.cs에서 gameObject를 받아와 Command 내의 private checkColor 함수

를 구현 당시에 초기화를 잘못 해주어 계속 같은 색깔로만 변했었음.

2) Wall2Command.cs에서 코드를 잘못 이해한 부분이 있어 클릭한 hit.point를 계속해서 받아왔으나, 해당 정보가 불필요함을 인지하고 후에 제외시키고 코드를 정리함.

3) TimeTable을 관리하지 않아 시간과 상관없이 Command들이 Replay될 때 일정 시간마다 역재생 되었으나, 후에 Time table 관리하는 코드를 포함 시켜 시간 차이를 두고 동작하였을 때, 이에 맞게 출력되게 변경함.

4. 구현 결과



1) 일반벽은 처음 의도대로 돌아가려 하고, 파괴할 수 있는 벽은 계속해서 부딪히려고 하기 때문에 꼭 삭제시켜줘야 함. 때문에 최단 루트를 위해 어떤 벽을 부수고 남겨둬야 할지 판단하면서 게임을 플레이하면 됨

5. 패턴을 적용했을 때와 안 했을 때의 차이점