2021.2학기 서강대 게임교육원 프로젝트

DNOTB (Do Not Open This Box)

프로그래머 작업일지

작성자: 이주화

최종 업데이트 날짜: 2021.12.08

자동목차

[<2021.09.08> 2](#_Toc89883233)

[<2021.09.13> 3](#_Toc89883234)

[<09.14> 4](#_Toc89883235)

[<09.15> 4](#_Toc89883236)

[<09.23> 5](#_Toc89883237)

[<09.27> 6](#_Toc89883238)

[<2021.09.28> 7](#_Toc89883239)

[<2021.10.01> 8](#_Toc89883240)

[<2021.10.05> 9](#_Toc89883241)

[<2021.10.08> 9](#_Toc89883242)

[<2021.10.18> 10](#_Toc89883243)

[<2021.10.20> 11](#_Toc89883244)

[<2021.10.21> 12](#_Toc89883245)

[<2021.10.26> 14](#_Toc89883246)

[<2021.10.27> 14](#_Toc89883247)

[<2021.11.01> 15](#_Toc89883248)

[<2021.11.02> 16](#_Toc89883249)

[<2021.11.03> 17](#_Toc89883250)

[<2021.11.05> 18](#_Toc89883251)

[<2021.11.07> 19](#_Toc89883252)

[<2021.11.12> 20](#_Toc89883253)

[<2021.11.13> 20](#_Toc89883254)

[<2021.11.15> 21](#_Toc89883255)

[<2021.11.16> 21](#_Toc89883256)

[<2021.11.17> 22](#_Toc89883257)

[<2021.11.21> 23](#_Toc89883258)

[<2021.11.22> 23](#_Toc89883259)

[<2021.11.24~25> 24](#_Toc89883260)

[<2021.11.27> 26](#_Toc89883261)

[<2021.11.28> 27](#_Toc89883262)

[<2021.11.30> 28](#_Toc89883263)

[<2021.12.04> 31](#_Toc89883264)

[<2021.12.05> 33](#_Toc89883265)

[<2021.12.07> 35](#_Toc89883266)

[<2021.12.08> 36](#_Toc89883267)

[<미마감, 또는 포기> 37](#_Toc89883268)

# <2021.09.08>

1. 각 씬 제작
   1. Logo씬 (=게임 시작 후 종료까지 영구 유지되야 할 매니저급 스크립트들 세팅 씬)
   2. Lobby씬
   3. Ending List Page씬
   4. Main Game씬

1-2: 각 씬의 UI 구성은 임시로 만든 것. 다 더미라고 보면 됨.

1. 각 씬으로 넘어가는 것을 관리하는 매니저 코드 생성
2. 추후 오브젝트 위에서 마우스 클릭(Down & Up)이 이뤄졌는지 체크하기 위한 부모 코드 제작 완료. 각 오브젝트들은 이를 상속 후 실행 부분만 조정해서 사용할 예정.
3. Main Game에서 이벤트 관리할 ‘이벤트 관리자’ 코드의 틀만 잡아 둠.

# <2021.09.13>

1. Main Game씬 일부 수정
   1. 더미 상자 오브젝트를 이용해서 엔딩 01~02 제작 후 동작 확인함.
2. 스크립트 변경점
   1. Box와 관련된 이벤트들 처리할 Box 스크립트 제작
      1. 테이프 관련해서도 이 스크립트서 관리
   2. 기존의 EventManager를 MainGame 국지가 아닌 전역으로 옮기고, 메인 게임 내에서만 활동해야 하는 이벤트 관리의 경우엔 SubEventManager 라고 새로 제작한 코드로 기능을 옮기고 연결함.
   3. 세이브 기능 추가
      1. 윈도우즈 기준 세이브 파일 경로: C:/User/User(=사용자명)/Appdata(숨김폴더)/LocalLow/SogandEduGames/DNOTB
         1. 현재 파일명: ‘DNOTB\_save\_event.csv
      2. 위에는 이벤트 통과만 체크할 생각. 이를 통해서 이벤트 통과 기록을 체크할 수 있게 되었으며, 관련 함수 제작 완료.
         1. 엔딩 3 이후부터의 엔딩에 대한 준비책. 그러나 아직 시작은 하지 않음.

# <09.14>

1. 게임 전반의 커서 시스템 추가
   1. Logo씬에선 변경 가능하나 하지 않음.
   2. 로비씬에서 불러들여 변경.
2. 3번째 엔딩 완성
   1. 관련 시스템으로 추가된 것이 위의 커서 시스템(칼날 커서용)
   2. 나이프 오브젝트 더미 추가
   3. 테이프와 송장 오브젝트 더미 추가
   4. 엔딩 달성 수를 코드 내부서 ‘깊이’로 표현하여 깊이 1 이상이어야 활성화되는 오브젝트들의 활성화 및 리셋 기능 관리 추가함. (주로 SubEventManager)
   5. 깊이 0 -> 깊이 1 활성화 때 조건부 버그 존재.
      1. 엔딩을 본 후 로비에서 들어오면 문제없지만, 리셋 테이프로 리셋할 경우 버그가 있음.
         1. 추후 엔딩 본 후 로비로 되돌리는 방식을 택하면 문제없어짐.
      2. 버그 내용: 나이프는 문제없이 활성화되는데, 스티커랑 송장이 활성화되지 않음. 리셋 테이프로 한번 더 리셋 할 경우 문제없이 복원됨.

# <09.15>

1. 코드 수정 작업 진행함.
   1. 처음에 만들어 둔 마우스 클릭 방식은 유니티 내부 시스템 이용을 줄인 채 사용했었는데, 크게 의미가 없는데다 오히려 일이 더 귀찮아진 스타일임을 깨닫고 유니티 기준으로 수정함.
      1. 예시
         1. 기존: leftup 포지션의 오브젝트와 rightdown 위치의 오브젝트의 위치값과 마우스 입력 포지션을 비교해서 가상의 네모박스 안에 클릭된 것이면 Execute 실행. (수정본 설명서 Execute 상세 설명)
         2. 수정: 유니티의 Raycast2D 및 Physics2D, tag, Collider 등 및 interface를 적극 활용해서, IEventObject 인터페이스를 포함한 스크립트를 태그&콜라이더 있는 오브젝트에서 뽑아내서는 Execute()실행. 이 Execute는 각 스크립트서 작성하므로 모두 로직이 다름.
            1. 기본 상속 구조 및 시스템을 변경한 것이므로 나름 대공사가 이뤄졌으나 기존 작성한 로직은 그대로 활용한데다 아직 초반이라 크게 공사 이뤄진건 없음.
            2. 그냥 움직임이 좀 깔끔해짐.
   2. 14일날에 보고한 버그 관련해선 이 수정과 함께 고쳐진 것으로 추정됨. 어느새 멀쩡하게 동작함.
2. 04 엔딩(포탈 엔딩) 플로우 제작 완료
   1. 큐브 더미, 큐브 포탈 더미, 큐브 엔딩용 스티커 더미 추가
   2. 큐브를 웨이포인트로 이동시키는 기능 추가
   3. 칼을 들고서 스티커를 건드리면 엔딩03, 칼을 건들지 않고 건드리면 큐브가 생성되며 엔딩 04 준비가 들어 감. 동시에 준비 가능함.

# <09.23>

1. 나온 오브젝트 리소스 적용
   1. 상자 Boxing, Untaping, Opened
      1. 테이프는 별도로 나올 것으로 생각하고 더미 작업했었기에 리소스에 맞춰서 코드 일부 수정함.
   2. 테이블
      1. 가로 크기가 미묘하게 안 맞아서 가로 세로 1.01 비율로 늘려서 적용함.
2. 나온 UI 리소스 묶음서 커서와 관련된 UI들 적용함
   1. 다른 UI 리소스도 확인했으나 관련 들어가는 부분들이 미구현 또는 어느 구역에 들어가는지 불명확한 점 등이 있어서 프로젝트 내부에 넣는 것은 보류함.
   2. 동일한 이유로 보류된 것으로는 컷씬 이벤트와 관련되어 나온 그래픽 리소스들도 존재.

# <09.27>

1. 나온 리소스 일부를 적용
   1. 상자 내에서 나올 엔딩 오브젝트
      1. 관련해서 신규 오브젝트와 관리 코드를 mainBox 코드와 오브젝트에 추가함.
2. 엔딩 컷씬 등장을 관리하는 클래스 추가.
   1. 3초 대기 후 5초간 컷씬 등장 등의 조건을 따라서, 시작 대기 초와 지속시간 초, 그리고 보여줄 컷씬 이미지(내부에 들어갈 이미지만)를 받아서 동작하는 코루틴 함수를 통해 관리.
   2. 대화 이벤트와도 관련해서도 처리 코루틴 함수 작성 완료.
      1. 배경의 페이드 인, 아웃을 코루틴 함수 내부에서 관리.
      2. 오래된 말들은 상단의 대화창으로 복사하여 밀어내면서 맨 아래의 대화창에 신규 말들을 입력하는 방식으로 구현됨.
         1. 동시에 보이는 말 풍선의 수는 3
      3. 모든 대화가 끝나면 다시 페이드 인을 한 후 위의 코루틴 함수를 부름.
         1. 이는 적절한 대화를 처리하기 위해서 엔딩 인덱스를 인자로 받아서 처리하는 부분을 활용한 것.
      4. 현재는 대화 부분을 인스펙터 뷰에서 노출하여 입력받고 있지만, 추후 save 파일 형태로 변경하여 관리하는 것이 더 편할 경우 해당 방식으로 변경할 예정.
   3. 다른 외부 함수서 엔딩 조건을 달성하면 이 클래스의 엔딩 인덱스를 인자로 받아서 적절하게 윗 컷씬&대화창 이벤트 관리 코루틴을 실행하도록 하는 함수를 부르는 방식.
   4. 리셋 테이프가 눌리면 모든 코루틴을 즉시 중단하고 관련 오브젝트들을 비활성화, 변수들을 초기값으로 돌려놓음으로써 즉시 초기 상태로 되돌림.
      1. 이는 ‘리셋 테이프 우선? 이벤트 진행 우선?’에 대해서 질답이 이뤄진 후 결정된 부분.
3. 윗 컷씬 클래스를 이용해서 1번 엔딩과 2번 엔딩과 관련하여 연출까지 완성함.
   1. 2번 엔딩과 관련해서는 관련 대사집이 진행되어 있지 않아서 더미로 기능 테스트만 진행함.
4. 일부 엔딩 조건 등에 대해서 인스펙터 뷰 노출 후 관리할 수 있도록 하는 것이 필요하다고 들어 조정함.
   1. 엔딩02 조건의 멍하니 있어야 할 시간에 대한 변수를 노출시킴. (Box 오브젝트의 MainBox 스크립트)
   2. 윗 엔딩 컷씬 관리 클래스에서의 대화창에 나올 텍스트와 텍스트간 넘어갈 속도 등을 관리하는 부분, 페이드 인 아웃 때의 불투명도에 대한 변수 등도 노출되어 있음. (SubEventManager란 오브젝트)

# <2021.09.28>

1. 일부 오브젝트 리소스 나온 것으로 대체
   1. 타임캡슐 리소스를 완성품 적용
   2. 커터칼 오브젝트 리소스를 완성품 적용.
   3. 송장 커버 스티커를 완성품 적용.
      1. 파일명이 Obj\_Invoice라고 되어있어서, 기존의 송장 오브젝트를 Obj\_Invoice로 변경하고 이번 받은 커버 스티커를 Obj\_Invoice\_Cover라고 수정하여 적용함.
         1. 사전에서 송장을 검색한 결과, Invoice가 ‘송장, 청구서’라고 나와서 사용하고 있지만 정확한 지칭 용어인진 모름.
2. 엔딩 컷씬 확인 리스트 기능 작성 완료
   1. 서브 카메라에서 렌더 타겟 텍스처를 통해서 텍스처를 뽑아내고, 그걸 Raw 이미지로 가공.
      1. 관련해서 UI 부분이 버벅였는데, 캔버스 타겟을 Screen에서 Camera로 변경하곤 Sorting Order 정리하여 해결
   2. 윗 유사 게임씬을 위해서 기존 게임 씬의 오브젝트를 복사해서 ListSceneBoxMain이라는 BoxMain을 상속한 코드를 만듦.
      1. 하는 작업은 달라도 들어갈 변수는 거의 같아서 상속함.
   3. 대부분은 B같이 기능을 따로 만들었지만, 엔딩 컷씬과 회화 부분은 동일한 코드 활용.
      1. 이리 되면 엔딩마다의 데이터(엔딩 컷씬 스프라이트, 대화 텍스트 등)를 여기서 입력받거나 하는데 이걸 2회 반복해야 하는 문제점 발생.
      2. 따라서 EndCutSceneDataManager 라고 관련 변수만 관리하는 스크립트를 별도로 만들어서 DDOL(게임 내 유지될 오브젝트)에 붙임.
         1. (소소한 프로그래밍적 기능 고민) 중복되는 그냥 매니저들에서 데이터 부분을 독립시킨 후, 그 부분을 프로퍼티의 get을 통해 관리하게 했는데, 다른 함수의 변수를 get으로 계속 읽어들이는 건 중간에서 주고받을 때 임시 변수 데이터가 생기면서 가비지가 늘지 않을까? 유니티 가비지 컬렉터를 좀 불러쓰게 되지 않을까?
            1. 그렇다고 각 코드별로 변수 만들고 읽어 들여 쓰자니, 중복되는 변수 데이터가 3개가 있는 꼴이 됨.
            2. 일단은 수정없이 현재 방식 유지.
   4. 동작 테스트 결과 문제없음.

# <2021.10.01>

1. 회의서 나온 화면에 그림자를 그림마다 개별로 할지 아니면 프로그램 기술로 처리할지에 대한 부분에 대해서, 프로그래밍 파트 부분의 프로토타입 제작 완료.
   1. 쉐이더를 손코딩해보다가 쉐이더가 기본 3D라 2D쪽으로 재가공하는 부분서 막힌 것을, 2020년도부터 유니티 제공의 커스텀 RP와 함께 제공된 비주얼 쉐이더 스크립팅 기술을 통해 제작
      1. 해당 손 코드도 그냥 3D의 Quad 등에 찍어봤을 땐 문제없이 나왔었음.
   2. 게임 오브젝트들은 따로 있고, 그걸 찍은 카메라의 결과물을 RenderTexture로 받아서 거기에 윗 쉐이딩 코드를 적용하면서 마테리얼로 재가공해서 뽑아내고, 이걸 RawImage 형태의 UI로 뽑아내선 덮어씌우는 방식인데, UI가 레이캐스트를 방해해서 기존 UI보다 Sorting Order가 낮고 레이 캐스팅 관련 컴포넌트를 제거한 서브 Canvas를 만들어 배치함.
   3. 우리 게임이 어몽어스처럼 움직이면서 화면에 그림자 보정 등 넣는 등이었다면 이 그림자 보정 구역을 계속 업데이트하면서 쉐이딩 하도록 하는 등으로 유효 활용 가능한 기술이겠다만은, ‘고정 카메라의 2D’란 점에서, 결국은 그림자용으로 레이어를 나눈 것의 각 픽셀의 색은 별도 지정 후 Alpha값을 RGBA값으로 재가공하여 만들어서 활용하는 기술임을 뒤늦게 깨달음.
      1. 윗 부분도 이미 유니티서 2D 라이팅 기술로 대부분은 만들어 둠. 사람마다의 개별 라이팅 연출도 결국 ‘Target Sorting Layers’로 해결 가능.
2. 회의서 요청받은, 현재의 엔딩 리스트 씬에서 엔딩 확인하는 장면의 크기 조정 및 적용되는 이미지를 윗 프로토타입 마테리얼로 변경.

# <2021.10.05>

1. 마테리얼 방식 -> 텍스처 방식으로 되돌려 놓고, 일단 오브젝트를 2개로 나눠서 마테리얼 방식과 텍스처 방식 어느 것으로도 가능하도록은 나눠둠.
2. 엔딩 1에 엔딩 3용 엔딩 오브젝트를 넣어두고 사용하고 있었음을 뒤늦게 깨닫고 수정함.
3. 엔딩 03 컷씬이 나와서 간이로나마 넣어둠. (01 엔딩과 동일하게 컷씬 등장)
   1. 아직 03 엔딩 연출 관련한 기획 문서/작업 문서는 확인하지 못함.
4. 엔딩 리스트 씬에서 일부 씬 재현 관련해서 덜 채워둔 부분 파악하고 수정함.
   1. 엔딩 씬 연출하는 부분의 로컬 함수로 UnBoxing과 UnsetCube 함수를 추가함. 관련한 오브젝트들의 활성/비활성화 세팅 함수.
   2. 컷씬 예시 이미지를 EndCutSceneDataManager에 등록된 값을 읽어들이는 것으로 바꿔도 충분한 형태가 되었는데도 이전에 만든 씬 매니저 객체의 구동을 별도로 변수 만들어서 활용하고 있는 옛 방식 그대로 쓰고 있었음을 깨닫고 수정함.
5. 박스가 열렸을 때 커터칼 부분과 그림이 맞붙는데, sorting order 설정 오류로 이상하게 보이는 것을 확인하여 수정함.

# <2021.10.08>

1. UI 이미지 리소스들 적용하면서 일부 갱신
   1. (공통)
      1. UI 이미지 치고는 여백이 너무 많은 이미지 형태로 줘서 그림판 3D의 자르기 기능 이용해서 모든 그림 그림 여백을 줄여봄.
         1. 그러면서 가능하면 4의 배율 사이즈로 맞춰서 유니티 압축 경고 안 뜨도록 변경.
      2. (커서 이미지 외) 유니티 내 UI 배치 및 사이즈도 적절하게 조절해둠.
         1. 엔딩 리스트 씬은 이미지를 ‘250 x 200’(0.5배율)의 크기로 들어갈 수 있도록 만들고는 그에 맞춰서 프레임 크기를 조정함. 가로 값 307, 세로 값 308 정도로 적절 값이 됨.
         2. 홈 버튼도 그냥 화면 비율에 맞춰서 채워 넣던 방식(메인 게임 기준, 가로 0.95 ~ 1.0. 세로 0.9 ~ 1.0)이었던 것을 ‘가로 세로 80 픽셀’이란 고정 방식으로 변경하면서 그에 맞춰서 앵커 값을 변경하는 방식으로 조정함.
   2. 나이프 커서 이미지를 새로 나온 사이즈 갱신판으로 적용함.
   3. 회화창 리소스 적용
      1. 관련 시스템으로 컷씬 관련한 시스템 일부 수정함.
         1. 눈에 보이는걸로는 Logo씬의 EndCutSceneDataManager에서 회화 설정하는 부분이 조금 바뀜.
            1. isLeft란 체크 박스를 on 하면 왼쪽 기준의 대화 풍선으로 바뀜.
            2. 대화 풍선 이미지 등록 부분도 생겨있음. 잘못 등록된 것이면 여기서 수정하면 됨.
   4. 로비로 돌아가는 버튼을 ‘홈 버튼’ UI 이미지로 바꿔 적용함

# <2021.10.18>

1. .서브 퍼즐 프로토타입들 제작 시작
   1. 전체적인 창 구조 제작
      1. World 타입 UI Canvas 오브젝트로 만듦.
      2. 창 위주로 관리하는
      3. 창의 왼쪽 위, 오른쪽 아래의 2가지 위치 값을 활용해서 마우스 클릭이 창의 범위 밖에서 이뤄졌다면 창을 종료하는 방식으로 창을 끔.
         1. 키는 것은 관련 오브젝트가 아직 없어서 버튼을 상단 UI로 추가함.
      4. 켜서 서브 이벤트 진행하는데 계속 엔딩 2번이 달성되어 버려서 서브 퍼즐 창이 켜져 있는 사이에는 시간 초 누적이 되지 않도록 수정.
   2. 다이얼 락 퍼즐 제작
      1. 4가지 다이얼에, 상하 버튼 눌러서 업 다운하는 구조.
      2. DialLockEventManager에서 모든 4가지 다이얼이 다 맞춰졌는지 체크하여서 맞춰졌다면 SubPuzzleManager에 신호를 줘서 다이얼 락 종료 & 서브 창 종료를 시킴.
      3. 각 DialLock 스크립트를 포함한 오브젝트들이 매칭되는 업다운 버튼의 실행 함수를 담당하면서, 눌리면 다이얼 번호가 돌아가고, 설정된 다이얼 값과 일치하는지 돌아갈 때 마다 매번 체크하여 그 결과 값을 DialLockEventManager 클래스의 자신과 매칭되는 변수에 할당함.
   3. Concentration 퍼즐 제작. (‘카드 매칭’, ‘신경쇠약’ 등으로 불리는 카드 게임)
      1. 5쌍(총 10장)의 카드
      2. 누르면 0.5초 후 뒤집히고, 두 번째 카드가 골라져서 다 뒤집어지기 전까지 다음 카드 선택할 수 없는 상태로 만듦.
         1. 이 두번째 카드도 다 뒤집어진 타이밍에 두 카드의 값을 비교하여서 같으면 매칭 됬으니 카드를 그대로 두고, 같지 않으면 매칭이 안 된 것이니 다시 뒤집음.
      3. 처음에 각 랜덤한 카드 위치에 ‘너는 이 숫자의 카드다’하고 지정해주는 방식을 랜덤한 값을 하나 얻어온 다음 그 값으로 넣을 수 있는지 체크, 못 넣는다면 다시 랜덤한 값을 얻는 것부터 다시 시작하는, 완료할 때까지 못 나오는 무한 반복문으로 간단하게 만들었더니 GC(쓰레기 수집자) 활동량이 미쳐버리는 등으로 프로그램이 얼어 붙는 문제가 생겨서 수정 진행함.
         1. order(카드의 값 순서 1쌍째는 1, 등으로 5쌍까지의 1~5값) 에 대한 List<int> 리스트 배열과, cnt에 대한 int[] 배열을 만들어서, (0,orderList.Count)의 범위 안에서 랜덤한 인덱스 값을 얻어서 이 값을 통해 order List를 조회하여, 이 값을 통해 적절한 값을 넣고, cnt값을 늘리며, 이 늘어난 cnt 값이 2이상(0,1,2.. 다보니 2 이상이면 ‘3개째’를 의미)이 되었다면 해당 order를 리스트에서 제외하는 방식으로 수정함.
         2. 당연히 값 넣기를 시작하기 전에 리스트의 상태와 cnt 값들은 초기화함.

# <2021.10.20>

1. 서브 퍼즐 프로토 타입 제작 2일차
   1. 슬라이딩 퍼즐(그림 조각 밀어 맞추기) 제작 완료.
      1. 조각 버튼의 상하좌우에 투명한 사각 이미지(이미지를 넣지 않으면 유니티서 포인터를 인식하지 못함)를 두어서, 퍼즐 조각을 선택해서 드래그하는 상태로 해당 이미지에 충돌하면 이동 시도.
         1. 이동하려는 곳이 비어 있다면(isHad 변수) 그 위치의 오브젝트의 자식으로 들어간 다음 로컬 포지션을 0이 되도록 이동하여 그 위치로 이동함.
      2. 이동 실패하면 부르르 떨리도록 만듦.
      3. 큰 구조는 EventManager(전체적인 조각 정보 및 조각 생성 및 초기화, 클리어 체크 내용 수합 등), Space(위치 체크 및 부모 오브젝트 관리용), Piece(각 조각의 UI 내용 및 각 클릭 및 움직임 관리, 개별 조각의 클리어 체크 등)의 3부분으로 나눠 적용함.
   2. 슬라이딩 퍼즐 관련해서 버그가 발생하여서 원인 추적 시도해 봄.
      1. (현상) 퍼즐 조각을 옮기다 보면 갑자기 한 조각이 엉뚱한 데로 이동함. 그러나 위치 간 자식 설정은 문제없음. 로컬 포지션 변경이 이상하게 이뤄지는 것.
      2. (해결책A) 일단 바로 해결책으로 R키를 누르면 로컬 포지션을 강제로 0으로 만들어서 순간이동 식으로 재배치하도록 하고, 이동 처리 시도 후 1초 후(더 줄일 수 있음)에 체크하여 로컬 포지션 값이 0이 아니라면 0으로 만들어서 옮겨버리는 방식을 적용해 둠.
      3. (원인1): 어떤 조각 (가)가 포인트 A로 이동 중인데, 다른 조각 (나)도 포인트 A로 이동하게끔 긁을 경우 (나) 조각이 엉뚱한 위치로 이동한 후 부르르 떨림. (발생률 높음)
      4. (원인2): 어쩐 조각 (가)가 포인트 A에서 B로 이동 중인데, 다른 조각 (나)를 곧 빌 포인트인 포인트 A로 이동시키려 할 경우 (나) 조각이 엉뚱한 위치로 이동함. (발생률 낮음)
      5. 이를 통한 (해결책2). 현재는 이동 관리를 각 Piece가 개별적으로 하는데, 모든 피스를 일괄 관리할 수 있는 EventManager 객체에서 이동 정보를 받아서 모든 조각들에 잠시 동안 이동 못하게 지연 시간 설정하도록 만들면 됨.
         1. 조작감이 좀 나빠지는 것도 있고, 관련해서 코드를 좀 더 수정해야 할 부분도 있어서 회의 등으로 확인 후 진행 예정.

# <2021.10.21>

1. 슬라이딩 퍼즐 디버깅 완료
   1. 엉뚱한 곳을 원인으로 보고 체크하고 있었던 것.
   2. 제대로 된 원인은 ‘Object Shaking’ 클래스 코드였음. 오브젝트가 부르르 떨려야 할 때 사용하는 기능.
      1. 흔들려랴- 하고 함수를 호출 받으면 지정된 시간초 동안 부르르 떤 후 원래 위치로 돌아옴.
   3. 발견하여 수정한 오류들 (주로 옛날에 쓴 코드를 가져와 쓰면서, 거기서는 고려하지 않아도 괜찮았던 부분들로 인한 오류였음)
      1. 시간초 관리 오류.
         1. 입력 받아서 설정 받는 초 시간과 실제 초 값은 별도로 관리해서, 함수 내 변수로 값을 복사해서 한 다음 이 함수내 변수를 시간초 줄임으로 써야 했는데 별도로 하지 않아서 한번 떨린 후 다시 안 떨리는 문제가 발생하고 있었음.
      2. 초기 위치 값 갱신 오류
         1. 떨리는 사이에 움직임 값이 움직이다 보니 초기 위치 값을 돌아올 위치로 가지는데, 오브젝트가 움직임에 따라서 이걸 또 갱신해야 했는데, 이 갱신하는 부분을 고려하지 못 했었음.
         2. 또한 그렇다 보니 떨리는 사이에 움직여 버리면 떨리기 시작한 옛 위치로 돌아가려고 하니까 문제가 생기는 것.
            1. 이것을 해결하기 위해서 떨리는 시간 초 동안은 선택되지 않도록 변경함. 선택되지 않으니 당연히 움직이지도 못함.
   4. 위의 ‘C(발견하여 수정한 오류들)’ 부분 수정이 끝나면서 어제 발견한 슬라이딩 퍼즐 버그 난 부분이 수정되어 깔끔하게 돌아가게 됨.
   5. 또한 마찬가지로 오류 원인을 잘못 파악해서 엉뚱한 함수와 변수들을 만들어서 해결한 부분이 있는데, 원인을 제대로 추적해서 해결한 후에도 그 함수 방식을 수정하지 않았던 것이 있었으며, 해당 부분을 수정해서 원래의 간결한 방식으로 돌려 둠. 테스트 결과 문제없음. (초기화 때 부모 설정하는 부분과 관련된 부분이었음.)
2. 슬라이딩 퍼즐 일부 코드 삭제
   1. R키나 1초 딜레이 시간 후 로컬 위치 값을 0으로 맞춰주던 기능 제거.

# <2021.10.26>

1. 일부 디버깅 등 수정 진행
   1. 모든 퍼즐의 EventManager 클래스를 싱글턴 패턴으로 변경.
   2. Concentration 퍼즐과 관련해서 모든 카드 10장에서 랜덤으로 설정하는 것이 아닌, 윗 카드 5장, 아래 카드 5장으로 해서 각각 매칭되는 수 1개씩 집어넣는 방식으로 수정 요청이 들어와서 해당 방식으로 수정함.
   3. Sliding Puzzle 과 관련해서 가장 처음 실행했을 때 퍼즐 프리팹 오브젝트 생성 후 배치를 제대로 못 하고 있는 점을 확인하여서, 이 초기 세팅을 하는 부분을 본인의 Start 나 Awake 이벤트에서가 아닌 상위 객체인 SubPuzzleObjects의 Awake 타이밍에 실행하도록 수정함.
      1. 문제없이 구동되는 것 확인함.
2. 파이프라인 퍼즐 제작 완료
   1. 버튼을 클릭해서 선택하면 뒷 배경의 빨간 하이라이트가 들어와서 선택됨을 알림.
   2. 다른 버튼이 선택되거나 버튼이 다시 눌리면 선택이 풀림.
   3. 선택이 된 상태로 하단의 반시계, 시계 방향을 지정하는 버튼을 클릭하면 해당 방향으로 90도 회전함.
   4. 각 파이프라인(way\_n\_n)에는 0, 90, 180, 270도 지정이 가능한 부분이 인스펙터 뷰에 노출이 되어 있고 이 리스트 박스에서 하나를 선택하면 이 파이프를 해당 방향으로 맞추면 정답 취급.
      1. 모든 파이프라인 조각이 올바른 방향으로 설정되면 게임 엔드.
      2. 현재는 4(기존은 0으로 했으나 0도와 180도 구분이 안되어서 변경)라면 0도, 1이라면 90도, 2라면 180도, 3이라면 270으로 해서 테스트 모드 맞춰 둠.
   5. 문제없이 구동되는 것 확인함.

# <2021.10.27>

1. 그래픽 리소스 적용
   1. 엔드 04 컷씬 리소스 적용
   2. 박스 상자 오브젝트 크기 조정된 것 적용하면서 위치 조정함
   3. 엔드 04 오브젝트 리소스들 적용함.
      1. 인박스 오브젝트도 별도로 추가 만들어서 적용
         1. 관련해서 기존에 BoxMain 스크립트에서 관리하던 InBoxObject의 그림을 InBoxImageManager라고 별도로 만든 관리자를 통해서 관리하도록 변경함.
   4. 리셋 테이프 오브젝트 리소스 적용함.
2. 엔딩 리스트 씬 수정
   1. 이전, 다음 페이지 개념을 추가하여 적용.
      1. 각 엔딩 리스트 버튼은, 각 버튼 마다의 0, 1, 2, 3 인덱스 + ((현재 페이지) \* (한 페이지 당 엔딩 수[현재 4]))의 인덱스에 해당하는 엔딩을 체크함.
   2. 엔딩 리스트 씬에서 BoxMain을 상속시켜서 수정하는 방식으로 리스트 씬의 상자 관리자를 만들어 사용했더니, 서브 퍼즐을 작성한 이후부터 오류가 나고 있었음을 뒤늦게 깨닫고는 수정함.
      1. BoxMain과의 상속 관계를 아예 품. 추가되기는커녕 오히려 줄어들다보니 상속 관계면 이외에도 손해가 상당했음.
      2. 수정 후 동작 테스트 결과, 문제없음을 확인함.

# <2021.11.01>

1. 서브 퍼즐 오브젝트를 월드 -> 스크린 스페이스의 캔버스로 이동.
   1. 월드 기준 캔버스를 했던 것이 일부 기능 사용에서 문제가 발생하여서였으나, 지금까지 짜면서 ‘이게 스크린 스페이스에서 안 돌아갈 리가 없는데?’ 싶어서 다시 수정해 봄 문제없음.
      1. 이전 에러는 아마 휴먼 에러. 저번 학기 것 참고하면서 변경해야 할 부분을 안 변경하고 가져온 것.
   2. 또 다른 변경 이유로는, 월드 기준의 UI로는 픽셀 단위로의 이미지 크기 설정도 안 되는 주제에 이미지 크기에 맞춘 객체가 생성되는 것도 아닌, UI랍시고 들어온 이미지를 강제로 자기에 맞춰서 재배열하는 문제 때문에 ‘가로 x 픽셀, 세로 y 픽셀’ 같은 식의 크기 지정 방식으로 하려고 하니 문제가 발생했기 때문.
2. 윗 캔버스의 이사와 더불어서 각 객체들의 크기와 위치를 조정해두고, 일부 연출을 추가할 예정이었으나 주어진 자료에서 ‘각 요소 간의 거리’는 있어도 어느 하나라도 ‘테두리로부터 ~정도 거리를 두고’나 ‘~위치의 요소를 기준으로 하여’ 같은 기준점을 잡을 부분이 덜 설정되어 있어서 대기 모드.

# <2021.11.02>

1. Sliding Puzzle(조각 맞추기 퍼즐)에 대해서 추가 연출 기능 추가.
   1. 이 연출을 위해서는 정확한 위치 값 및 이동해야 할 거리를 알아야 하였다 보니 대기하고 있었으나, 현재 얘만은 지금 자료로 충분함을 깨닫고 진행함.
      1. 3\*3 구성에, UI 공간 중앙에 배치된 것이 보이는 예시 구성이니, 0.5 피벗 기준으로 중앙 피스를 (0,0,0)으로 두고, 각 조각들을 조각의 크기(125. 정확히는 각 조각의 절반크기인 62.5 가 2개 각각 더해지면서 원래 크기가 된 것.) + 각 조각 간의 거리 값을 활용해서 각 위치 값을 잡아주고, 이동 거리는 각 조각 간의 거리 값을 이용하면 됨.
      2. 하지만 다른 퍼즐들의 UI 배치들에 관려해서는 여전히 기준점이 될 놈을 가늠하기 힘듦. 자료 보충이 필요함.
   2. 만들어서 테스트해 본 결과 문제없이 구동됨을 확인함.
2. 전날 캔버스 이사하면서 일부 필요 없어진 기존 객체나 일부 변수를 그대로 남겨둔 것을 깨닫고 수정함.
3. 빌드 테스트 후 일부 문제점 발견하여 수정 진행.
   1. 기본적으로 전체화면 모드로 실행됨.
      1. 1280, 720의 창모드 실행으로 변경. 또한 전체 화면 변경 및 창 크기 수정은 불가능하도록 함.
   2. 회화창이 처음 나올 때 객체를 초기화하지 않아서 활성화해둔 더미 상태로 보이는 문제점이 있었음.
      1. 관련해서 처리하는 부분서 관련 코드를 추가하는 것을 잊었던 것뿐. 휴먼 에러였음. 수정함.
   3. 메인 게임 창에서는 문제가 없는데 엔딩 리스트 씬에서 체크할 때 컷씬과 관련된 UI들이 크기 조정이 이상하게 됨.
      1. 1280, 720이라는 프로그램 해상도를 제대로 인지 못하고, 디바이스의 1920, 1080을 인식하고서는 거기에 맞춰서 서브 카메라 기준의 서브 캔버스가 확대된 것이라고 보임.
      2. 서브 캔버스를 카메라 해상도 기준으로 스케일링 하는 것이 아닌 스케일링 없는 고정적인 픽셀 값으로 수정했더니 문제가 없어짐. 어차피 서브 캔버스는 서브 카메라가 찍는 화면이고, 서브 카메라도 고정적인 화면만 찍다 보니 화면 변경에 따라서 크게 문제가 생기지 않을 것이라 추정됨.
4. 2D 라이트 연출이 추가될 것이 예고됨에 따라서 유니티 URP 시스템을 조금 만져보면서 2D 라이팅이 추가될 수 있는 환경을 찾아서 설정함.
   1. 단독으로 왜 있는건지 모르겠었던 2D RP를 URP의 인스펙터서 대상 RP 등록하는 곳에 덮어씌우니 됬음.

# <2021.11.03>

1. 리소스 적용
   1. Concentration 미니 게임(카드 뒤집어서 짝 찾기 게임) 관련
      1. 카드 뒷면과 앞면 나온 것 적용함.
         1. 앞면 적용과 관련해서는 추가적으로 코드 작성하고 기존 텍스트로 표시하던 코드를 제거함.
         2. 회전 애니메이션이나 회전중임을 알리는 텍스트가 없으니 위화감이 들어서 뒷면에서 앞면으로 변경될 때에는 회전 대기 시간을 적용하지 않음. (뒷면으로 되돌아가는 것은 여전히 해당 딜레이가 존재함.)
         3. 배경 나온 것도 적용함.
            1. 관련해서 어색해진 카드 위치 값은 여전히 관련 정보 대기 중이므로 아직 수정하지 않음.
   2. Dial Lock 미니 게임(회전 다이얼 맞추기)과 관련
      1. 배경과 버튼 이미지는 적용 완료함.
      2. 같이 제공된 파이프라인 퍼즐에서의 회전 버튼은 회전을 표현한다고 보기 힘들어서 적용 보류.
   3. 로비 씬에서 나올 타이틀 텍스트 이미지도 적용해 둠.

# <2021.11.05>

1. 디버깅
   1. (보고받음) 상자를 열고서 리셋했는데 상자가 제대로 리셋이 안되었다.
      1. 체크해본 결과, ‘열었다-‘라고 처리하면서 이미지를 변경하고 있는데 리셋이 진행되서 그런 것. 리셋되서 닫힌 이미지로 변경했는데, 처리하고 있던 열었다 이미지로 변경이 일어나는 등의 순서가 꼬인 것.
      2. 회의 때는 이건 어떻게 조절할 수가 없다고 했는데, 생각해보니 이벤트 큐 방식이면 나름 조절할 수 있음을 뒤늦게 깨닫고 국지적으로 이벤트 큐 방식을 채용해서 처리하도록 변경함.
         1. 크게 차이는 없을 건데, 외부 함수서 부르는 타이밍들끼리 꼬이는 것이랑, 외부 함수서 부르면 그걸 자기의 이벤트 큐에 넣어서 이벤트 큐 순서대로 불러서 처리하는 것이 되니까 순서 꼬임은 덜해지지 않을까 기대. (검증 힘듦)
            1. 만약 이게 안된다면 그냥 한 공정을 더 넣음으로써 성능 저하만 했을 뿐. 심각한 저하는 아니겠지만.
   2. (보고받음) 서브 퍼즐 중인데 엔딩 2가 나왔다.
      1. 이미 서브 퍼즐이 진행중이면 엔딩2 조건용 시간 초 체크를 들어가지 않도록 해 놓았음.
      2. 체크해보니 엔딩2 조건이 달성된 그 순간에 서브 퍼즐을 클릭한 듯 했음. 타이밍 문제.
      3. 엔딩 조건이 달성되었다면 서브 퍼즐창을 닫도록 변경함.
   3. 윗 B 버그(서브 퍼즐 중 엔딩2)를 체크하기 위해 하던 중, 오히려 엔딩을 이미 봤는데 엔딩2도 나오는 문제를 발견하여 수정함.
      1. 이미 엔딩을 봤다면 관련해서 오브젝트들의 충돌체들은 비활성화 시키는 처리를 해 둬서 문제가 없었는데, 시간초 처리는 해 두지 않았음.
      2. 엔딩 컷씬 관리하는 스크립트서 엔딩이 켜졌는지 체크하여 이 값을 통해 시간초 체크 진입 조건을 추가하여 해결함.
   4. (보고받음) 카드 짝 맞추기(Concentration 카드) 게임 중에 이미 뽑은 카드를 한 번 더 누르면 또 뽑히는 기능적 하자가 있음을 보고 받음.
      1. 그저 휴먼 에러. 수정하는 코드를 넣음.
2. 위치 조정 정보가 추가적으로 제공되어서 위치를 조정함.
   1. 다이얼 락 배치 수정 완료
      1. 받은 가로 간격이용해서 활용.
      2. 세로는 (0,0) 기준.
   2. Concentration (카드) 완료
      1. 가로는 중앙(0,0) 기준.
      2. 세로는 따로 계산해서 간격 받아서 계산.
   3. 파이프라인 수정 완료.
      1. 파이프라인은 0.5, 0.5의 중점 지점으로 계산하기 힘들어져서 제공된 자료서 [파이프라인 영역(회색)]이라 표현된 것을 0,1(왼쪽 위) 기준으로 해서 거리를 두게 만듦. 당연히 이미지 피벗도 0.1로 주고 진행.
      2. 내부 조각들은 회색 영역의 0.5, 0.5 기준. 버튼들은 1.0, 0.5 기준에 피벗도 0, 0.5로 수정해서 사각영역 바깥 부분에서 진행으로 됨.

# <2021.11.07>

1. 리소스 적용
   1. Concentration 카드 게임에서 쓰일 앞면 이미지가 새로 수정된 것이 있어서 변경하여 적용함.
2. Concentration 관련해서 디버깅
   1. 05일날에 넣은 코드가 ‘덜 넣었고’ ‘엉뚱한데 넣었음’을 뒤늦게 깨닫아서 수정함.
      1. 카드가 매칭되면 비활성화하도록 해둠. 아마 보고받았을 때 ‘매칭되서 종료된 카드를 또 뽑을 수 있더라’ 라는 식으로 보고받은 것으로 여기에 넣었다고 봄.
   2. 카드가 돌아가면 다시 선택되지 않도록 변경하였으며, 카드가 다시 뒤집히면 활성화하도록 함.
      1. 카드가 매칭 되었을 때와 열렸을 때(즉 앞면이 보일 때들)에는 카드가 선택되지 않도록 함.
      2. 카드가 매칭을 체크하여 ok면 바로 선택 가능하도록 하고, 틀려서 다시 뒷면으로 되돌아가고 있는 딜레이 타임 도중이라면 모든 카드의 버튼을 비활성화해서 조작되지 않도록 변경함.
         1. 일부 디버깅 해보다가 이게 더 조작감이 좋고, 오류를 방지할 수 있어서 이리 변경. (이미 조작된 카드 선택이나, 변경되고 있는데 계속 누를 경우 오류가 나는 부분이 있었음)
      3. 카드가 매칭되면 위의 ‘모든 카드의 버튼 기능 비활성화’와 ‘모든 카드의 버튼 기능 활성화’에 관련된 이벤트 객체와의 연결을 끊어서 오작동을 방지하며, 이 연결이 끊겨있는지 아닌지를 bool 변수로 체크하여 리셋 때 필요하면 재등록, 필요하지 않으면 상태를 갱신하지 않고 넘어가는 처리를 추가함.

# <2021.11.12>

1. 리소스 적용
   1. 각 버튼에 공통적으로 사용된다는 UI 버튼 오브젝트가 있어서 적용함.
      1. 유니티 UI Button의 기본 형태인, ‘버튼 배경 이미지’ + 별도의 텍스트 컴포넌트 조합 타입.
2. 다이얼로그 시스템을 리소스에 맞춰서 재수정함.
   1. 기존은 왼쪽, 오른쪽 발화자를 구분하는 방식이었으나 이젠 그냥 발화자를 표시하는 부분이 추가된 단일 타입이 되어서 해당 방식에 맞게 짜둔 코드를 수정함.
   2. 엔딩 02 다이얼로그 적용함.
   3. 엔딩 03에도 다이얼로그가 추가된 것을 확인하고 시스템 수정하여 추가함.

# <2021.11.13>

1. 2D 라이트를 통한 그림자 적용
   1. 그림자 길이 조절 기능이 있는 듯이 매뉴얼은 정리가 되어 있었는데, 없음. 매뉴얼에 나와있는 해당 속성도 없음.
      1. 인터넷 검색해보니 2D 라이트의 그림자 조절과 관련해서는 토론이 많았고, 공식 유니티 답변도 ‘이거 필요하다고는 생각하는데 그림자 코드를 거의 전체적으로 수정해야 해서요. 필요한 기능 부분들에 대해서는 답변해 드릴게요’가 달리고는 그 아래로는 모두가 나도 이거 수정도전했다- 난 이리 해봤다, 그리고 실패했다로 포럼이 있었음.
   2. 계속 검색하고 여러 조합 도전해보고 배치해보고 조정하고 반복하다가 포기해야겠다 싶어질 쯤에 왠지 돌아가는 조합을 발견해서 적용. 무난하게 적용됨.
      1. 한 오브젝트에 ‘라이트2D’와 ‘쉐도우캐스터2D’를 같이 배치하고, 모양을 조정한 후 라이트에서 그림자 파워 조정. 쉐도우 캐스터는 Self Shadow 설정. 그리고 모양 조정하면 빛 모양 설정한 모양으로 그림자가 그려짐.
      2. 단, 모드는 ‘Additive’가 아닌 ‘Alpha Blend’ 모드로 해야 함. (2020까지는 Additive 고정. 2021.1로 프로젝트를 업그레이드 함.)
   3. …일단 적용은 했는데 유니티의 2D 라이트와 2D 그림자가 2D 게임에서의 시야 거리 조절이나 일부 부분 밝혀지는 것, 오브젝트에서 노멀맵과 조합하여 예쁘게 만드는 용도 등 맵 전체적인 작용에나 적절하지, 오브젝트 배치와 상호작용으로 생각할 부분은 아님을 새삼 배움.
      1. 대표적으론 어몽어스 같은 게임의 시야 조절.
      2. 빛끼리 겹치면 색이 괴랄해진다는 점도 있어서.. 하아..
      3. 애초에 이거 사용하는 순간에 상당히 무거워진다는, 상당히 비효율적인 시스템인 모양에, 애초에 오브젝트 별의 2D 그림자를 별도로 뽑아내면 조합해서 사용해서 끝나는 일인데, 통짜 레이어로 만들어서 오브젝트별로 그림자를 분리해낼 수 없다 & 그림자 적용 파트를 지정하는 마스크 텍스처를 별도로 만들 수 없다..라고 반복되어서 진행한 것.. 시스템적으로도 필요 없는 부분이고, 다시는 하고 싶지 않은 생고생.

# <2021.11.15>

1. 이전 2D 라이트를 통한 그림자를 적용할 때, 상자 온 오픈에 따른 칼날이 가려지는 그림자 등은 조절하였었으나 게임 진행하면서 칼을 집었을 때 칼이 사라짐에 따라 그림자가 사라져야 했음에도 불구하고 이를 지우지 못했던 것을 뒤늦게 깨닫고 수정함.

# <2021.11.16>

1. 나온 리소스 적용. (애니메이션 3종 + 알파) (각 적용점에 맞게 코드도 수정함. 애니메이션 재생 후 원래하던 작동 시작)
   1. 로비 타이틀 배경이 메인 게임 시작 시 상자가 열리는 애니메이션 형식으로 나와서 적용함.
      1. 게임 플레이 누르면 1회 실행되고 종료.
   2. 메인 게임 씬에서 2개의 애니메이션 추가
      1. 송장을 덮는 커버가 벗겨질 때의 애니메이션이 추가되서 추가함.
      2. 큐브 엔딩 마지막에 버튼 누를 때, 버튼이 들어가는 애니메이션이 추가되서 추가함.
         1. 4프레임으로 내려가는 곳까지의 애니메이션을, 7프레임으로해서 버튼이 눌리고 되돌아오는 곳까지로 수정함.
   3. 모든 애니메이션은 실행시간 1초로 잡아 둠. 이는 필요에 따라 조정 가능함. (전체 프레임 수를 조정한 후 각 키 값을 적절한 프레임으로 옮겨주면 됨.)
   4. 회전 버튼 리소스 적용함.
      1. 나오긴 예전에 나왔는데 있는 것을 확인하는 것도 늦었고, 적용해야 하는 파이프라인 퍼즐과 관련한 리소스도 진행이 안되어 있었어서 적용을 보류했음. 이번 적용에 들어서 추가적으로 적용해본 것.
2. 리소스를 적용하면서 큐브 오브젝트 실행하는데 위치가 이상해졌음을 깨닫고 수정함. (아마 기존에 상자 그림을 새 상자 그림으로 변경하면서 미묘하게 위치 수정하면서 꼬인 것)

# <2021.11.17>

1. Invoice Cover(송장 커버)의 애니메이션 실행 후 리셋을 하면 커버가 제대로 복원되지 않는 버그를 발견하고 수정함.
   1. 원인은 애니메이션이 진행되면서 마지막에 다 벗긴 이미지로 변경된 상태에서 기존 상태로 돌려주지 못한 것 때문.
   2. 기존 상태에 대한 스프라이트 정보를 추가적으로 받아서 처음 활성화될 때 해당 스프라이트로 교체하는 방식으로 수정하여서 문제를 해결함.
      1. 오브젝트의 활성화, 비활성화는 이미 BoxMain이라는 실질적인 씬의 이벤트 관리자 객체에서 진행하고 있음.

# <2021.11.21>

1. 기존 2D Light 활용한 그림자 제거. 마테리얼도 Unlit 형으로 다시 재설정.
   1. 대신에 나온 그림자 오브젝트들 적용.
   2. 그림자를 적용하면서, 너무 커서 줄여 두었던 리셋 테이프가 박스에 기존 크기로 영향을 주고 있어서 기존 크기로 돌림.
      1. 열렸을 때 테이프가 이상하게 보이는 점이 생겨서 코드를 통해서 닫혔을 때와 열렸을 때의 sorting order의 값을 조절하는 부분을 추가함.
         1. 닫혔을 때 2, 열렸을 때 4. (리셋 테이프는 3)
   3. 화면 전체 그림자는 알파 없이 쓰자니 너무 진해서 128/255의 알파 값을 적용함.
2. Sliding Puzzle 미니게임 쪽에 나온 리소스 적용함.
   1. 기존 텍스트 표시에서 스프라이트 표시로 변경
   2. 원래는 각 피스에서 마지막 위치 이동 후 1초 뒤에 종료하도록 했는데, 이젠 이벤트 매니저에서 딜레이 값을 인스펙터 뷰에서 입력 받아서, 해당 초만큼 완료 이미지를 띄운 채 대기한 후 종료하는 방식으로 변경함. (현재 설정값: 2초)

# <2021.11.22>

1. 기존 EventManager를 해체하고 그 기능을 DataRWManager와 MainEventManager(기존 SubEventManager)에 배분함.
   1. 엔딩 깬 개수는 DataRWManager를 참고하여 읽어 들인 부분이다보니 바로 그쪽으로 이동시킴.
   2. EventManager의 기능을 실질 메인 게임에선 기존의 SubEventManager, 리스트 씬에서는 ListSceneBoxMain이 담당하고 있고, 이리 하는 것이 괜스레 코드 등록 및 해제로 꼬이는 것을 방지할 수 있다고 보아서 해제하고 연결되어 있던 모든 코드들을 수정함.
   3. 얼마 전부터 코드 해설 문서를 작성하기 시작하면서 발견한 수정점.
2. 컴퓨터 반응 속도가 느려진 상태로 Sliding Puzzle을 시연해보니, Pointer Collider의 민감도가 상당히 낮아서 슬라이딩 조작 인식을 제대로 못하는 문제점을 발견하여 수정함.
   1. 기존은 항상 활성화에, 선택됨을 bool 값으로 저장하여, 콜라이더 안에 포인터가 들어왔을 때 이 값이 true이면 이동 조작을 하였고, 콜라이더 크기는 그림 외곽에서 +0.05 수준이었음.
      1. 이를 +0.2수준까지 올려도 느려서 그냥 +1.0 수준으로 올림.
      2. 포인터 콜라이더를 애초에 선택되면 활성화, 선택되지 않으면 비활성화 시키는 방식으로 변경함.
      3. 체크해본 결과 문제없이 동작됨을 확인.

# <2021.11.24~25>

1. 유니티 글리치 쉐이더 제작
   1. 노이즈 효과를 원한다길래 글리치 쉐이더들 검색해서, 옛날에 나온 오픈 소스 쉐이더 코드를 보고 Unity URP의 쉐이더 그래프로 해당 쉐이더를 제작해봄.
      1. Kino Glitch란 것.
         1. [GitHub - keijiro/KinoGlitch: Video glitch effects for Unity](https://github.com/keijiro/KinoGlitch)
         2. 아날로그와 디지털 글리치가 있는데, 디지털 글리치는 기본 텍스처 외에 글리치용 텍스처가 별도로 필요해서 제외함.
      2. 원하는 효과라면서 예시로 받았던 듀토리얼 동영상의 쉐이더도 같이 제작해둠. 사인 효과 및 밝기 효과까지 추가된 부분은 제외.
         1. [Horror TV Glitch Shader (Creepy) | Unity Shader Graph Tutorial Deutsch - YouTube](https://www.youtube.com/watch?v=Tt07sfAVj7E)
   2. 기존 회의 때 디더링하기 위해서 나온 텍스처를 그대로 활용하는 것을 통해 화면 노이즈를 주는 것으로 착각해서 디더링 디더링 했는데, 보통 이런 연출은 ‘글리치’가 맞음을 뒤늦게 깨닫고 방향 꺾어 진행한 것.
2. 그림자 파일이 수정되어서 수정하여 적용 및 추가된 부분들에 맞춰서 코드도 일부 수정.
   1. Inbox 오브젝트 일부에도 그림자가 추가되서 관련 오브젝트 및 관리 코드 추가.
   2. 상자가 열렸을 때에 그림자가 추가되서 관련 오브젝트 및 관리 코드 추가.
      1. 위치 값 조절 및, 필요에 따라 크기 조절까지 해가며 자연스럽게 맞춰보려 했지만 일부 그림자가 테이프 오브젝트랑 좀 부자연스럽게 되는 점은 있음.
   3. 다른 부분은 수정된 것 적용하면서 관련 오브젝트 조금 추가 됬으나, 관리 코드로는 만질 부분이 없어서 넘어감.
3. 엔딩 01은 특수 연출, 그 외 컷씬 연출에 대해서 기획서에서 변경된 것을 확인하여서 관련 코드 수정함.
   1. 기존은 컷씬을 부르면 3초간 대기 후, 5초간 보여주고 종료되는 방식.
   2. 변경된 방식은 들어갈 때, 회화처럼 뒷 배경이 페이드 아웃되고, 페이드 아웃이 설정된 투명도(60%)까지 진행되고 나면 컷씬을 띄워 보여줌.
      1. 보여주는 시간 안내가 없어서 기존의 5초를 유지함.
      2. 회화 이벤트의 경우는 종료 후에 페이드 인 후 컷씬을 띄우는 방식이었는데, 기존 구조로 했더니 페이드 인 -> 페이드 아웃을 반복하여서 보기 안 좋다고 느껴져서 페이드 인 과정을 지워봤으며, 그 불투명도를 그대로 유지하면서 컷씬을 실행하도록 함.
         1. 회화가 끝나고나서 바로 컷씬을 띄우는 것을 체크하다보니, 마지막 글을 읽을 틈이 거의 없이 넘어감을 뒤늦게 깨닫아서 종료 처리에 앞서서 1초간 대기하는 딜레이 타임을 추가함.
   3. 특수 연출을 제외한 일반 연출 때는 컷씬을 보여준 후 다시 페이드인 하고서 컷 씬 관련한 오브젝트들을 정리하는 기능을 수행하도록 함.
   4. 01 엔딩에서의 특수 연출 때는 컷신을 보여준 후 다른 컷씬서의 페이드인 타이밍에 0.6(60%)의 불투명도에서 1.0f(100%) 불투명도까지 추가적으로 페이드 아웃이 진행되며, 이 페이드 아웃이 종료되면 그림의 색을 흰색에서 검은색으로 변경하여 블랙 아웃 느낌으로 변경(사운드가 추가되면 이 타이밍에 사운드)하고, 1초간 기다린 후 로비로 돌아감. (엔딩 리스트 씬에선 창만 종료함).
      1. 이 엔딩을 통해 로비로 돌아가는 경우엔 로비 씬에서 검은 이미지로 화면을 가려두고, 이걸 페이드 인함.
         1. 이 페이드 인이 진행 중일 때는 로비 씬에서 아무 동작도 되지 않도록 해둠. 연출이 끝나고 나서 정상적인 활동 가능.
      2. 01 엔딩의 컷씬을 부르는 타이밍에 들어서면 리셋 테이프의 콜라이더를 비활성화해서 클릭되지 않도록 하는 기능을 추가함.
         1. 다른 엔딩들과 달리 씬을 넘어가는 기능이 추가되면서 조정한 부분. 현재는 괜한 오류를 피하기 위해서 이 타이밍에 하지만, 컷씬 활성화 타이밍이나 두번째 페이드 아웃 시작 타이밍의 추가적인 두 타이밍으로 늦출 수도 있음.
4. 타이틀 로고 그림 텍스트가 수정된 것 적용함.

# <2021.11.27>

1. 사운드 매니저 제작
   1. 배경음, 환경음(동일 사운드 중복 재생 없는 효과음), 효과음(동일 사운드 중복 재생 있는 효과음)으로 구분한 것을 따라서 구현함.
      1. 환경음은 이벤트 큐 방식, 효과음은 오브젝트 풀 방식을 사용.
      2. 배경음은 루프 설정을 해둬서 로비씬 들어간 이후로는 끊기지 않으며, ~~효과음들은 클립 재생 시간을 얻어와서 딜레이 타임을 설정, 아니면 음악 클립 파일의 재생 길이 값만큼 재생&활성화한 후 비활성화.~~
         1. 오해하지 않게 보충. 효과음은 루프 설정 없음.
      3. (위에 취소선 이유) 위는 그냥 유니티 오디오 소스의 Play() 함수 운용만 생각한 것. PlayOneShot() 이란 함수가 추가적으로 있고, 이것으로 운용하면 오디오 클립마다 별도의 오디오 소스로 운용하기 위해서 오브젝트나 프리팹으로 구분할 필요가 없이, 하나의 오브젝트(하나의 오디오 소스)로만으로도 운용이 가능해서 해당 방식으로 변경.
         1. 왠지 clip.Length(=오디오 클립 완료 시간)의 값을 제대로 못 얻어온다는 문제점 때문에 좀 더 연구해보다 발견.
            1. 결국 이 클립 길이 값은, 파일마다 플레이 시간을 유니티 인스펙터 뷰에서 체크하여 해당 값을 수동으로 입력하는 방식으로 수정함.
         2. Loop 설정 등은 걸어줄 수 없지만, 현재 재생되고 있는 사운드와 동시에 특정 사운드 클립을 넣을 수 있어서 효과음 연출엔 최적인 함수. 0~(클립 완료)까지 자동으로 재생됨.
            1. 연속해서 부르면 이전 소리가 재생되고 있던 말던 겹쳐서 들리도록 하기 때문에, 다른 소리와는 몰라도 같은 소리의 중복 재생을 막는 쪽으로 운용하려면 여전히 이벤트 큐 방식은 유지해줄 필요가 있음. (환경음)
            2. 둘 다 상관 없는 클릭 등의 효과음은 그냥 바로 이 함수만 사용해서 호출됨.
      4. 엔딩 01 배경음 변경 요청은, 엔딩 04를 하고 있는 도중에도 잘못 나올 수 있고, 엔딩 03서도 잘못 나올 수 있겠다 싶은 타이밍에 요청을 하여서 설명 후 엔딩01로 확정된 후 변경하는 방식으로 구현함.
      5. 모든 씬에서 오디오 리스너를 지운 후, 이 사운드 매니저를 DDOL의 하위에 부착한 후 여기에 오디오 리스너를 붙임.
      6. Space Ship, Notification 효과음 관련해서는 아직 넣을 곳이 완성되지 않아서 보류.
         1. 각각 엔딩5, 신규 힌트용 시스템.
      7. Click은 리소스가 없어서 보류함.

# <2021.11.28>

1. 자잘한 요소들 수정.
   1. 카드 뒤집기 효과음을 플레이어 미조작시의, 실패한 카드를 다시 뒤집어 두는 동작에도 넣어두니 난잡하게 느껴져서 해당 부분을 제거. (의견 교환 결과)
   2. 특정 상황서 사운드 쪽의 실행 중인 특정 코루틴을 멈추는 코드를 짤 때 ‘!=null’ 설정을 하는 걸 잊어서 오류 메시지 띄우고 있던 부분을 ‘!= null’ 설정을 넣어서 뜨지 않도록 변경.
   3. 알람02 효과음 실행 시간 타이밍을 변경함. 기존은 컷씬 실행 종료 시에 나왔으나, 이제는 컷씬 등장 때 나옴.
   4. 클릭 사운드를 추가하고, 실행 코드도 추가함.
      1. 실행 타이밍도 SoundManager 클래스에서 관리함.
2. 배경 벽지 그림이 나와서 적용함.
   1. jpg지만 배경이라 문제없음.
   2. 다만 jpg라서 그런지 픽셀 크기는 기존 테이블과 크게 차이가 없는데(같진 않음), 기존 테이블은 크기 문제가 거의 없었는데(1.01, 1.01 스케일링하니 화면에 딱 맞는 수준), 이번 건 가로(x축)로 1.02, 세로(y축)로 0.63 스케일링 해줘야 화면에 맞는 크기가 됬음.
      1. (긴급) 이 부분에 대해서, 지금까지 오해한 채 작업이 진행되었을 수 있음을 깨닫고 대화 진행. …문제 제대로 인식했고 오해 맞았음. 모든 오브젝트 그림이 1280x720 기준으로 그려진 것이 아님. 개별 오브젝트의 사이즈를, 캔버스 사이즈 크기가 아닌 구도에 맞게 변형하기 전의 오브젝트 박스(=구도화 시키지 않은 오브젝트, 화면에 90도로 비춰지는 오브젝트에서의 그 화면)의 크기로 착각하고 작업을 진행하셨다고 함.

# <2021.11.30>

1. Magnifier(돋보기) 커서 및 이 커서 상태로 일부 오브젝트 클릭 시 설명문 나오도록 함.
   1. 커서 피벗 설정은 x,y 순으로 (0.23, 0.38)로 함. 그림판에서 돋보기 중앙점이 대충 어디쯤인가 체크해서 집어넣은 수치임.
      1. 왼쪽을 0으로 한 x축으론 23%(100%면 오른쪽), 위를 0으로 한 세로축으로는 38%(100%면 아래)쯤 위치라고 잡은 것.
   2. 설명문 중 Table은 관련 클릭 처리도 없고, 기획서에서도 어떤 메시지가 나올지 제대로 정리되어 있지 않아서 목록으론 있었으나 뺌.
      1. 그 외 설명문은 기획서대로 넣음.
   3. 설명문을 띄우는 리소스는 기존 회화 오브젝트를 재활용. 그래서 기존 회화 리소스를 관리하던 EndCutSceneManager와 연결되서 활용함.
   4. 이 설명문을 등록하고 관리하는 MagnifierManager 스크립트는 MainGame씬의 SubEventManager 오브젝트(윗 컷씬 매니저가 붙어 있는 오브젝트와 동일)에 붙어있음.
      1. 당연하지만 이벤트 씬에선 없음.
   5. (-A)기존 설명문이 나와있는 데 재차 설명문이 나올 클릭을 하였거나, (-B)설명문이 나와있는데 엔딩 02 조건이 달성되어서 회화문이 나와야 하는 타이밍이라면, 기존 실행 됬어야 할 회화문 종료 코루틴(설정된 지연 시간이 지나면 나와 있는 설명문을 지우는 작업 진행 시작시키는 코루틴)을 중지하고 새 설명문을 띄움. (-B)의 경우라면 지연시간 기다림없이 바로 지우고 초기화하곤 회화문 설정을 시작하는 것.
   6. ‘엔딩 03’ 플로우 차트를 보니 송장을 그냥 열었는가, 돋보기로 체크하여 미니게임을 진행하였는가로 엔딩 03-1, 03-2로 분기하는 방식으로 변경된 것을 확인하여, 위에 E로 설명한 코루틴의 타입을 나눠서 회화문 종료 후 Concentration(카드 게임) 미니 게임을 실행시키는 코드를 추가함.
      1. 설명문이 정상적으로 종료되기 전에, E의 종료 코루틴을 지우는 작업이 진행되면 퍼즐도 자연스레 실행이 되지 않는 구조임. 변경은 가능함.
      2. 미니 게임이 실행되면 자동으로 노멀 커서로 변경하는 기능 추가.
   7. 돋보기 버튼(120, 90 크기)을 우측 하단의 위치로 설정함.
      1. 돌아가기 버튼(홈버튼)도 기획서상 위치가 지정되어 변경되어 있음을 깨닫고 우측 상단에서 좌측 상단으로 위치를 옮김.
      2. Magnifier Button이라는 전용 관리 코드를 만들어서 붙여 둠. 붙였을 때의 버튼이 실행할 일과, 버튼의 이미지의 색 관리를 수행. 변경될 색은 이 스크립트의 인스펙터 뷰 노출된 값을 통해 설정 가능. (RGB가 모두 동일한 값의, 무채색계의 값에서 조절해야 위화감이 적음. 아니면 단순 알파 값만 조정도 가능.)
2. 커서 매니저 일부 수정
   1. 기존에 노멀과 칼만 커서가 변경될 때는 커서 매니저에서 관리할 필요 없이 나이프 쪽에서 관리하면 되었지만, 이번 돋보기가 추가되면서 일일이 관리하는 식이면 코드 관리가 힘들어져서 구조 수정.
   2. 오브젝트와 연결된 커서들(나이프, 돋보기)에 관해서는 Unset Event라는 event 객체와 거기에 연결되는 각 함수를 만들어서 매칭. 커서가 변경되는 타이밍에 변경 전의 커서가 해당 커서였다면 해당 커서의 Unset event를 실행하도록 만듦.
      1. 나이프는 비활성화된 그림을 다시 활성화.
      2. 돋보기는 기존 살짝 변경된 그림을 원상태로 되돌림.
   3. 이 구조로 변경하면서 노멀 커서로 변경하는 코드를 Reset 이벤트(게임 첫 시작 & 리셋 테이프 눌렀을 때 등에 수행) 타이밍에 수행하도록 추가함. (MainEventManager에서 추가함. 즉, 메인게임에서만.)
3. 글리치 연출 관리자 코드 제작
   1. 글리치를 딜레이를 가진 채 / 아닌 채로, 입력된 시간 초 만큼 활성화 시키고 다시 비활성화시키는 함수들을 포함.
   2. 메인 게임에서와 엔딩 리스트 씬에서 작동법이 조금 달라서, 엔딩 리스트 씬과 관련해서는 매니저 코드에, 메인 게임에서의 처리는 글리치 스크린 자체에서 OnEnable, OnDisble로 동작을 관리하는 스크립트를 하나 새로 추가해서 덧붙임.
      1. 메인 게임서는 캔버스 설정을 실시간으로 변경해줄 필요가 있음.
      2. 엔딩 리스트 씬에서는 일반 카메라 텍스처 오브젝트와 글리치 포함된 카메라 텍스처 오브젝트를 한쪽만 활성화시킬 필요가 있음.
   3. 엔딩 03과 관련해서 일단 기획서에 있는 부분들에 연출 코드를 추가함.
      1. 송장 커버 제거 후, 송장을 2회 클릭 시, 글리치 스크린 1초 활성화
         1. 애초에 송장은 여러 번 클릭하는 시스템이 없었는데, 어려운 건 아니라 기존 커버 여러 번 클릭하는 것을 참고해서 기능을 추가 작성함.
            1. 2회로 글리치 스크린, 3회 이상으로 완전히 벗겨짐.
            2. 위 ‘횟수’ 인식은 돋보기 커서면 가산되지 않음.
         2. 미니 게임(Concentration) 클리어 시 1초 기다린 뒤 0.5초간 글리치 스크린 활성화.
         3. 다이얼로그, 컷신 팝업 종료 후 글리치 스크린을 1초간 활성화. & [분기 03-1 역시 마찬가지로 다이얼로그 종료 후 글리치 스크린을 활성화한다]…
            1. 일단 뭔 소린지 명확하진 않은데, 03-2면 다이얼로그랑 컷씬이랑 각각 종료 후에 1초씩, 03-1이면 다이얼로그 종료 후에 활성화. (컷씬 보일 땐 항상 짧게 글리치 스크린이 적용된다고 보면 됨)
            2. 컷씬 종료 후 글리치 타이밍은 ‘페이즈 인’ 시작 타이밍.
4. BoxMain에 있던 ‘MyEnding’ 네임 스페이스를 EndCutSceneManager로 옮김.
5. 엔딩 03을 엔딩 03\_1과 엔딩 03\_2로 분리시킴.
   1. 대화문 관련해서는 기획서에 나온 것 적용.
   2. 컷씬 이미지는 양측 모두 똑같이 적용함.
      1. 대화문을 보니 둘 다 큰 차이가 없는 듯 보임.
   3. 세이브 데이터도 03을 03\_1, 03\_2로 변경. 기존 04 자리를 03\_2가 대체하게 됨.
      1. 추가적으로 세이브 데이터와 관련해서 ReadDataRW(Logo 씬의 DDOL 오브젝트에 붙은 스크립트)의 인스펙터에서 ‘Save Reset’이라는 체크형 변수를 노출시킴. 이걸 on하면 기존 세이브 데이터를 초기 세이브 데이터로 다시 덮어씌울 것임.
6. 이벤트 해지 코드들에 관해서, 이벤트 관리자가 null 상태(이미 먼저 파괴된 상태)라면 해지를 넘어가도록 하는 코드 추가.
   1. 종료 시점이라 에러가 났다해도 문제가 적고 실질 에러도 아닌 부분이지만, 로그가 계속 남는 게 거슬려서 추가함.
   2. 관련해서 Knife 커서 이벤트를 신규 등록하면서 해지하는 부분 추가하는 것을 잊어서 게임이 반복되고 씬이 전환되고 복귀함에 따라 잘못된 이벤트 등록 유지 & 실행으로 인해 오류가 발생한 것을 뒤늦게 깨닫고 수정함.
7. 엔드 컷씬 매니저의 ‘회화문’ 띄우는 구조를 조금 변경함.
   1. 실제 사용적으론 변경된거 없음. 그냥 코드 구조적으로 변경.
   2. Switch Case 문을 1개로 하되, 매번 중복되는 부분이 있어도 개별 복사해서 개별 변경했던 것을, Switch Case 문을 2개로 늘리되 복사되는 부분(회화문 처리 부분)은 첫번째 Switch Case서 가져온 값을 같은 변수에 집어넣어 처리하고, 두번째 Switch Case 문에서 회화문 처리가 끝난 다음에 개별 처리하는 부분들을 처리하도록 함.
8. 그래픽 리소스 업데이트 된 것 적용.
   1. 퍼즐 리소스 하나 추가됬다는데 어디 올라왔는 지 안 보여서 적용 X.
   2. 배경 리소스가 조금 수정되서 나와서 변경 적용함.

# <2021.12.04>

1. 사운드 조절 효과가 추가될 것으로 보여서 관련 코드 수정.
   1. 유니티 오디오 플레이어를 단일로 만들어뒀던 것을 배경음용 & 효과음용 컨텐츠를 나누고 코드로도 연결할 수 있도록 하고 수정함.
      1. 루프 설정도 코드 단계에서 조정하도록 함. 인스펙터뷰에서도 설정은 해뒀는데 실질 의미는 없음. 배경음은 루프, 효과음은 루프 x.
   2. 설정된 옵션 값을 저장할 파일 ‘DNOTB\_save\_option.csv’를 추가하고 관련해서 저장 코드를 추가적으로 만짐.
      1. int로 저장하다보니 float 0~1.0f의 값을 \*100 한 값을 Round(반올림)한 값을 저장하는 방식으로 만듦.
         1. 즉, 0~100의 값으로 사운드 값이 저장됨.
2. 연기 애니메이션 추가
   1. 인박스 오브젝트 아래에 추가. 활성화 코드는 코루틴 함수 & 호출 함수를 별도로 만들었으며, 활성화 전 지연 시간초를 입력 받으면 그 시간 동안 대기 후 1초간 재생하고 종료하도록 함.
   2. 윗 함수 실행 타이밍 및 관리는 서브 퍼즐 매니저에서 카드 게임 종료 타이밍 때 관리할 수 있다면 그게 낫지만, 괜히 꼬일 수 있어서 박스 관리자에서 호출하도록 함.
      1. 따라서 이 부분 연출 조정할 때면(앞선 글리치 실행시간), 서브 퍼즐 매니저와 박스 매니저를 둘 다 관리해야 함.
      2. 3\_2 엔딩 컷씬 실행도 지연되서 실행되도록 하는 부분을 추가함.
3. 모든 코루틴 포함하는 인게임 오브젝트 관리 스크립트들에, 리셋 이벤트 타이밍 때 실행하던 코루틴 함수들을 모두 종료하도록하는 코드 추가함.
   1. 원래 추가했었어야 했던 부분을 잊고 있던 것.
4. 글리치 강도를 기획서에 정리된 대로 값을 입력함.
5. 그래픽 이미지 리소스 변경 및 추가
   1. UI 기존 리소스들 색감 달라진 수정버전 나와서 수정 버전 적용.
   2. 파이프라인 리소스 추가 적용 및 완성 시도.
      1. 해보려고 기획 문서를 체크하는데, 단순한 텍스트로만 설명이 되어 있고 좌표값 (2,2)와 (3,3)이 연결 된다 식으로, 대각선으로 연결된다고 되어 있어서 일부 고정 좌표값(십자모양) 외에는 각 조각의 값을 정확히 알기 힘들어서 문의 -> 사용법 공유하곤 직접 제작 해서 받기로 함.
      2. 받고나서 이하 작업들 이어함.
6. End04 수정 및 End05 제작.
   1. 하면서 만진 부분이 많아서 정리하기엔 너무 장문이 되어서 생략.
   2. End04 관련해서 CubeCol, CubeButtonCol(큐브와 큐브와 관련된 테이블 버튼의 클릭 인식하는 스크립트)이라는 두 스크립트에 관해서 대개편이 이뤄졌고, 관련해서도 BoxMain에도 추가된 함수 다수.
   3. 엔딩 05와 관련해서는 컷씬 리소스는 따로 나온 것이 없어서 별도로 슬라이딩 퍼즐 완성 이미지를 재활용함.
   4. 기존에 큐브 나오는 것이 송장 클릭과 관련되어 있었는데, 현재는 관련 흐름이 완전히 변경됨. 제거하는 것 잊고 테스트 해보다가 다시 마주하곤 관련 코드 제거함.
7. 슬라이딩 퍼즐 배경도 나와서 적용.
   1. 여분의 공간 없이 슬라이딩 퍼즐 공간만 존재하여 해당 부분과 관련한 스크린 보더를 추가하고, 이것을 슬라이딩 퍼즐의 on/off에 따라서 기본/추가 간에 변경되도록 함.
8. 엔딩 05의 엔딩 오브젝트들도 적용함.
   1. 진행 과정은 엔딩 04서 분기되는 것이라 문제가 없음.
9. 약간 디버깅 진행. 이번에 만든 것들 동작 테스트.

# <2021.12.05>

1. 엔딩 04 분기에서 05로 넘어가는 타이밍을 놓치면 더 이상 05 진입은 안되는데, 05 진행 중에 버튼을 잘못된 타이밍에 누르면 04로는 진입이 되어 버리는 문제(?)가 있어서 잠시 고민해 본 결과, 엔딩 05 루트 확정이 됬을 경우에 대한 bool 변수를 하나 더 만들어서 일반 엔딩 04용 버튼 클릭 횟수 인식하여 분기되는 부분들 자체를 05루트 확정시 진입하지 않도록 함.
   1. 엔딩 05 마지막에 버튼이 눌려져야 하는데, 이 부분은 위에 진입시키지 않은 부분과는 별도의 부분이라서 동작에 문제없음.
   2. 눌러서 반응을 해야하는데 눌리지 조차도 않는다(=애니메이션 실행도 안 된다)는 것 자체가 어느 의미 힌트가 되긴 했음. 04로만 테스트해봤던 결과.
2. 엔딩 리스트 씬 다듬기
   1. 배경 오브젝트 적용 안 되었던거 적용.
   2. 최종 엔딩 수는 6개인데 1페이지당 4개의 엔딩칸으로 해서 8개의 칸을 보여주고 있어서 이 부분을 다듬어서 1페이지는 4개, 2페이지는 2개 보여줄 수 있도록 다듬음.
      1. 문제 없이 진행됨을 확인함.
   3. 엔딩03-2의 연기 연출, 엔딩 05와 관련해서 코드 추가 등 진행함.
3. 서브 퍼즐 스킵 버튼 추가.
   1. 문제없이 동작 확인됨.
   2. 슬라이딩 퍼즐과 관련해서는 여분의 공간이 없어서 아슬아슬하게 윗 부분에 설정해 두었는데, 그래서 버튼 클릭이 아닌 화면 외곽 클릭으로 인식해서 동작 안하고 꺼지는 경우도 있음. 그렇지만 그렇다고 안쪽으로 넣기에는 실제 퍼즐 진행에 방해가 됨.
4. 페이즈 아웃 연출 때 일부 UI 순서가 잘못 정리되어서 페이즈 아웃 영향 안 받고 좀 거슬리게 보이던 부분이 있어서 UI 오브젝트 배치를 조절해서 순서를 재조정함.
5. 그 외 자잘한 미해결 버그 정리 시도
   1. 테이블이 콜라이더 관련 설정을 하였어도 조사 클릭이 안됨.
      1. 조금 비효율을 감안하고서도 레이캐스트 대상을 최초의 1개만이 아닌 모든 대상을 가져와서, 태그 비교(원래부터 했음)를 통과한 첫 대상만 처리하고는 넘어가도록 코드를 수정해봤지만 되지 않음.
      2. 레이 캐스트 발사 길이(=깊이) 부족인가 싶어서 값도 변경해봤는데 되지 않음.
      3. 체크해보던 결과, 얘 혼자 BoxCollider2D가 아닌 BoxCollider가 들어가 있었음을 뒤늦게 깨닫음.
         1. 수정하니 문제없어짐. 위의 코드 변화는 다시 제거.
   2. 위 테이블 문제랑 비슷하지만 좀 다른 문제로, 큐브랑 송장이 겹치면서, 큐브가 앞에 나와 있어서 이게 눌리어야 하는데 송장이 먼저 클릭 인식되는 문제가 있음.
      1. 겹치지 않는 큐브 상단 끝단을 누르면 문제가 없이 되긴 함.
      2. z 위치값을 아무리 조정해도 문제 발생. Sorting Order 문제는 더더욱이 아님.
      3. ‘큐브 관련 오브젝트들은 하나로 묶어 관리하기 위해서 계층을 하나 더 추가하여 관리하고 있었는데, 설마 같은 계층이 아니라서 좀 더 상위 계층인 송장이 우선되는 것인가?’하는 번뜩임이 있어서 관련해서 조금 수정해봄.
         1. 해결 완료.
   3. 일단 본인 테스트에서는 이 이상의 버그는 발견되지 못함. (글리치 관련해서는 위화감 있는 부분이 엔딩 05 하면서도 또 나왔는데, 이 부분에 대해서 예전 보고했을 때도 이걸로 문제없다고 들었으니 넘어 감. 어떤 의미로는 연출 중에 퍼즐이 진행되어도 곤란함.)
      1. 글리치 활성화됬을 때 일부 UI요소가 갑작스레 화면에서 사라짐. 옮기는 처리는 해놨고, 일부 UI는 제대로 옮겨 감. 이 사라지는 UI 중 대표가 퍼즐 종류. 글리치 중에 퍼즐이 진행되도 곤란하니 오히려 낫긴 함.
      2. 그렇지만 글리치 연출 중에 퍼즐이 활성화되는게 아닌, 글리치 후 퍼즐이 활성화되어야 하는데 글리치 연출이 진행 중(거의 막바지)에 활성화되는 건 오류라고 봐야 함.
         1. 글리치 연출을 1초 후 0.5초 + 1.5초 후 퍼즐 활성화.. 의 두 명령이 동시 진행인데, 퍼즐 활성화를 1.75초로 늘려 봄.
         2. 원인 자체를 잘못 파악했음을 깨닫고 추가 변수를 넣어 관리함.
            1. 퍼즐을 풀지 않고는 퍼즐 창을 꺼버렸을 때, 다시 퍼즐을 풀 수 있도록 2회 클릭 이상 상태인데 퍼즐이 클리어 되지 않았다면 퍼즐창을 다시 활성화해라고 해둔 상태였음.
            2. 위의 딜레이 활성화 + 이 2회 클릭 이상 인식이 동시에 진행되던 것.

딜레이 활성화가 실행되었는가에 대한 bool 변수를 하나 더 추가해서, 이 bool 값도 ok이고 2회 이상 클릭이다면으로 변경.

문제없이 실행됨을 확인함.

# <2021.12.07>

1. 일부 리소스 적용
   1. 연기 애니메이션이 색 변경되서 재제작 되었기에 Ver2로 만들어서 적용함.
   2. 큐브 빛나는 것 관련해서는 그래픽 선에서 할까, 2D 라이트로 할까 분명히 묻고 진행해서 2D 라이트의 단점을 얘기했음에도 2D 라이트로 진행하겠다고 결정됬는데 뒤늦게나마 그래픽 리소스 재등장.
2. 디버깅
   1. 유니티 Play 모드서의 테스트와 실제 빌드 파일서의 속도 차이(있을 때가 있고 없을 때가 있는)의 원인을 찾아 수정함.
      1. 원래 이동할 때 (Speed 값) 단일을 이용하는게 아니라 (Speed 값 \* Time.deltaTime)을 해줘야 모든 환경서 동일한 속도 값이 나오는데 이 기본적인 부분을 깜박하고 안 적용함.
   2. 엔딩 진행 중인데 돋보기로 오브젝트 클릭하면 종료되고 나오는 문제가 있어서 수정함.
      1. 이 부분에 대해서도 마찬가지로 여기까지 신경을 못 쓴 것. 시스템 만들 때 엔딩 진행중임을 알리는 변수는 이미 만들어 놓고는, 돋보기 때 엔딩 컷씬의 자료를 쓰면서도 이 변수를 체크하지 않음.
   3. 엔딩 리스트 씬에서 회상을 종료했음에도 사운드가 나는 부분이 있어서 관련 부분 처리를 추가함.
      1. 이 부분도 마찬가지로 테스트 부족했던 것.
      2. 사운드 이펙트만 종료되도록 처리함.
         1. 이 부분을 메인 게임에서의 리셋 이벤트에서도 처리하도록 추가함.
            1. ev\_reset 타이밍에서 실행되면 테이프 소리조차 멈추다보니, 이걸 실행하는 코드를 부르는 테이프 콜라이더 관리자(TapeReBoxing)에서 진행함.
      3. 엔딩 회상씬이 열리면 기본적으로 BG를 MainGame으로 바꾸도록 변경하고, 씬이 종료되면 다시 Lobby BG로 변경되도록하는 코드를 추가함.
3. 교수님 지적부분
   1. 상자 열리는 때에 버튼들 유지되는 부분을, 상자 열리는 애니메이션 진행 때 버튼들을 비활성화 시키는 연출 추가함.
   2. 대화창의 크기 및 글자 크기가 너무 작지 않은가 하여서 키움.
      1. 대화창 크기는 기존 가로 1100, 세로 100에서 세로 150으로 늘림.
         1. 배치도 y값 포지션 기준 0.1, 0.3, 0.5에서 0.15, 0.4, 0.65로 변경
      2. 헤더 텍스트 크기 40, 본문 텍스트 크기 20 적용.

# <2021.12.08>

1. 디버깅
   1. 빌드 파일 플레이 테스트를 할 일이 생겨서 하다가 일부 시스템 오류를 발견해서 디버깅.
   2. 테이블이 돋보기가 아닌데도 클릭되서 나오는 점을 수정
      1. 커서 타입 체크하는 부분이 빠져 있었음.
   3. 큐브 일부 이동이 여전히 속도가 이상함을 발견.
      1. 이전에 ‘Speed \* Time.deltaTime’으로 변경해주는 부분서 빼먹은 부분이 있었음.
   4. 커서가 돋보기인데 박스 테이프가 뜯기는 일이 벌어짐.
      1. 이 부분에서 커서 체크하는 부분이 빠져 있었음. 수정함.
   5. 일부 타이밍에 큐브를 클릭하면 쓸데없이 추가적으로 빛나는 경우도 발견했는데, 해당 부분 코드 수정 자체가 어려울 건 없지만 애초에 큰 영향도 없고 하니 수정 포기함.
      1. 엔딩이 확정된 타이밍에서 추가적으로 더 누를 경우에만 발생함. 이걸 >= 체크로 해두어서 그런 것임은 아는데, 굳이 ==로 바꾸고 ++ 요소를 넣어서 더 작동 안되게 하면 됨은 아는데 굳이 찾아 수정할 필요를 못 느낌. 애초에 허술한 부분이 많아서.
   6. 실질 게임에 활용되진 않았으나 테스트 기능은 해볼 수 있는 파이프라인 퍼즐에서 시계방향 회전 버튼이 제대로 동작하지 않는 문제점을 발견.
      1. 회전 그림 파일이 반시계방향 하나만 나와서 해당 이미지는 y축으로 180도 회전시킨 이미지 파일을 쓰고 있었는데, 이 탓에 버튼이 뒷면이 눌리고 있다고 판단되어서 제대로 인식을 못한 모양.
         1. 버튼이 아닌 이벤트 트리거를 써도 마찬가지로 클릭 인식을 못함.
      2. 이미지를 별도 하위 오브젝트로 분리해서 이걸 180도 돌리고, 이 녀석의 회전 값은 건드리지 않고 이벤트 트리거만 별도로 장착함.
         1. 버튼으로 해서 이미지 연결을 했을 땐 여전히 동작을 안함.
         2. 동작에 문제없음.
            1. 대신에 유니티 기본의 버튼 눌림에 따른 Color Tint 효과는 되지 않다 보니까, 기존에 잘 되던 역회전 방향 버튼에서도 이 효과 옵션은 제거함.

# <미마감, 또는 포기>

1. 사운드 조절용도의 Setting 팝업창 기능 추가
   1. 마감 D-3(12.04)일 때, 이런 기본적인 기능에 대한 기획 요구가 추가.
      1. 이 부분에 대해서는 오해였음은 밝혀짐. 몸 아파서 쉰 날 중에 재공지된 모양. D-day는 14였음. 그러나 이 부분에 대해서 분명히 얘기를 했는데도 오해 수정은 없었음.
   2. 12.04에 관련 시스템 넣을 준비는 함.
   3. 12.05 오후 경 리소스도 나옴.
      1. 그러나 세팅창 버튼용 이미지 등 일부 리소스는 자의적인건지 빠져있음.
   4. 역량이 아닌 감정적, 건강 상태적으로 더는 못 버티겠기에 포기함. 엔딩들에 대해서야 구상 정리가 늦게 된 것이라고 봐도, 이런 기본적인 시스템까지 이제서야 한다는 점 때문에 마지막까지 남아있던 기대, 믿음, 책임감 등이 다 깎여 나감.
2. 크레딧 기능 추가 요청
   1. 1-D랑 마찬가지. 이건 기본적 시스템 종류도 아니라서 넣을 필요성도 못 느끼겠는데, 이유를 물어도 컨텐츠 부족으로 감점 걱정하는 이유 정도 밖에 안 되었어서 본인 선에서 각하함.