Hell's DIEter 제작 보고서

2021년 1학기 게임제작프로젝트(2)

게임소프트웨어전공 B893248 정해빈

목차

개요

문서 개요

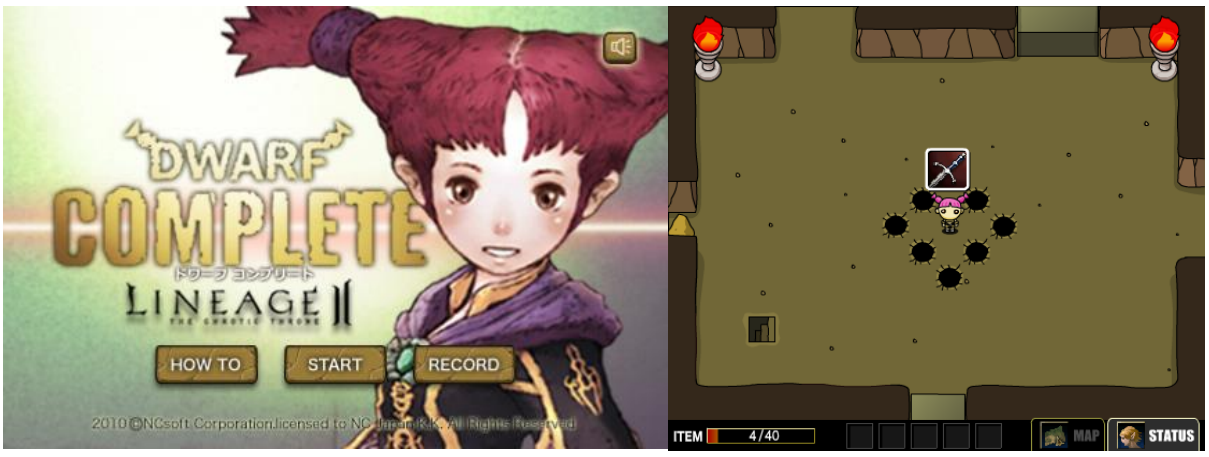
본 문서는 Hell's DIEter(이하 헬스 다이어터) 구현 과정 및 결과에 대한 내용을 담고 있습니다. 구현 계획, 구성도, 구현 내용 및 결과 순서로 기술하였습니다. 구현 계획에서는 구현 전 기존 게임에서 어떻게 구현했는지를 분석하는 과정을 포함했습니다. 구현 결과의 경우 시행착오를 겪으며 최종적으로 구현된 내용을 기술하는 것이 인과관계가 명확히 보일 것이라 생각하여 함께 기술하였으며, 이외의 기획관련 내용은 기획서에 서술하였음을 알립니다.

구현 계획

레퍼런스 게임 조사

기획 및 구현을 시작하기 앞서 기존 게임 중 구현하고자 하는 게임과 비슷한 컨셉을 참고하여 퍼즐, 어드벤처 장르의 게임들을 분석해봤습니다.

Dwarf Complete(드워프 컴플릿)



플래시 게임으로 만든 퍼즐 어드벤처 게임으로, 우주선을 타고 탈출하는 것이 이 게임의 목표입니다. 아이템을 줍고, 주운 아이템을 조합하여 도구를 만들거나 퍼즐을 풀어나가며 잠긴 문을 열며 나아가는 게임입니다.

이 게임에서 퍼즐을 풀어 잠긴 문을 여는 것과 아이템을 찾아 탈출하는 것을 참고하여 기획하였고 구현 계획을 세워봤습니다.

Little Nightmare(리틀 나이트메어)

3인칭 퍼즐 어드벤처 호러게임으로, 쫓아오는 추격자들을 피해 탈출하는 것을 목표로 하고 있습니다. 컨트롤에 강약조절이 필요한 점과 마찬가지로 퍼즐을 풀며 나아가야 하는 점을 참고하여 컨트롤이 어렵지 않은 게임을 만들고자 하였습니다.

게임 시퀀스별 플로우 차트

플레이어가 어떻게 행동해야 스테이지를 클리어할 수 있는지에 대한 클리어 조건을 세우기 위해 플레이어의 행동에 대한 플로우 차트를 작성해봤습니다.

튜토리얼 스테이지

4페이지의 왼쪽 그림은 튜토리얼 스테이지에서의 플레이어 행동에 대한 플로우 차트입니다. 튜토리얼 스테이지에서 이벤트는 코루틴과 delegate를 이용하여 플레이어가 해야 할 행동을 확인하고 다음 퀘스트를 진행하는 과정을 구현해봤습니다. 플레이어가 NPC에게 말을 걸면 퀘스트가 시작되며, NPC의 대사가 끝나고 퀘스트를 진행해야 수행확인을 하도록 설정했습니다.

라이트 유저를 대상으로 기획한 게임이기 때문에 튜토리얼은 간결하면서도 쉬운 내용으로 진행해야 한다고 생각했습니다. 사람에 따라 퀘스트 내용을 이해하지 못했거나 다시 보고싶을 수도 있다고 생각하여 모든 대화가 끝나고 NPC에게 다시 말을 걸면 퀘스트에 대한 내용을 다시 들을 수 있도록 설정했습니다.

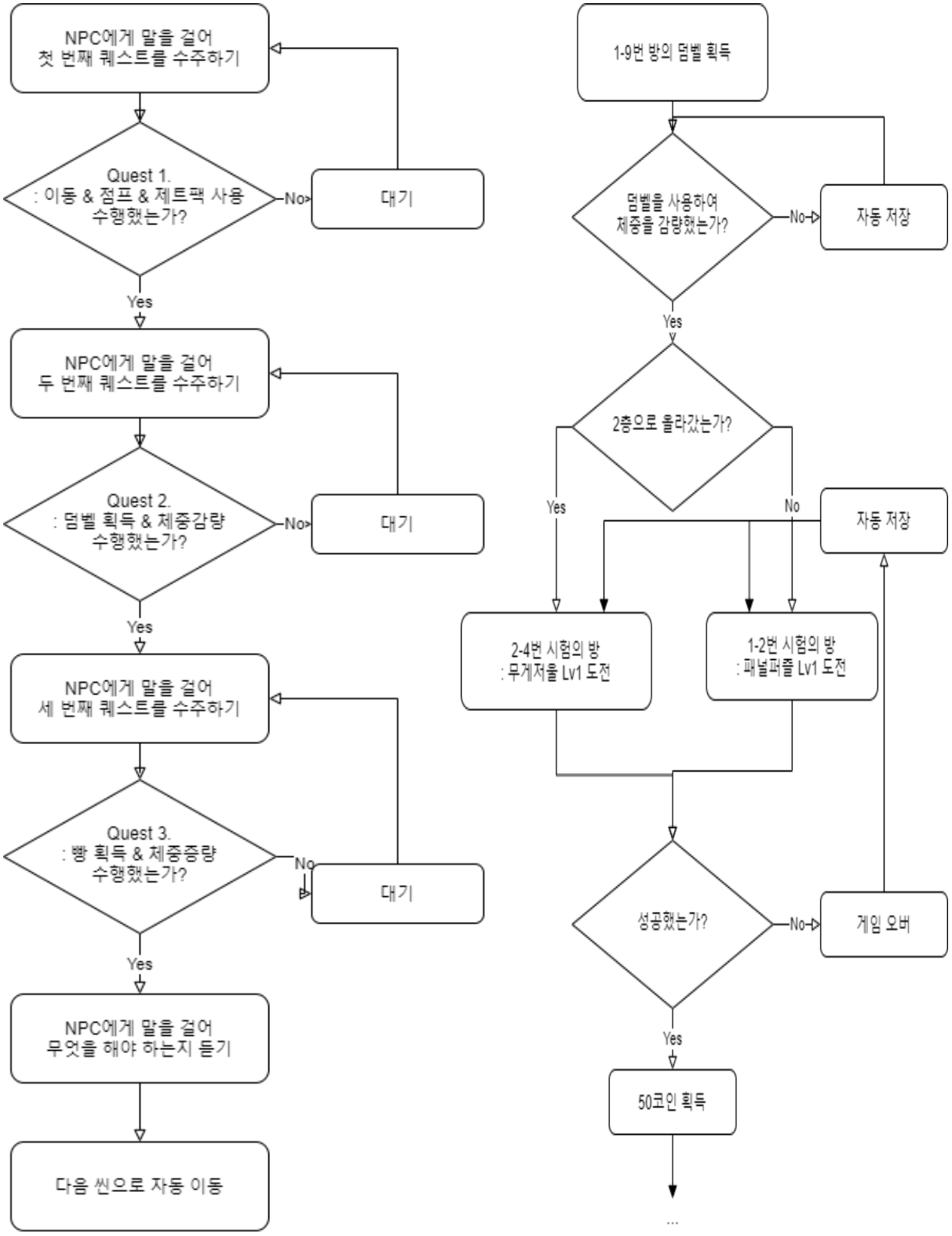
(대본 스크립트를 관리하고 Parsing하는 과정은 구현 내용에 후술하였습니다.)

메인 스테이지

메인 스테이지에서는 기획서에 서술한대로 시험의 방을 통해 아이템을 획득하거나 획득할 수 있는 능력을 갖추면서 진행해야 게임이 좀 더 쉬워지도록 디자인해봤습니다. 4페이지 오른쪽 그림은 첫 번째 시험의 방을 시도하고 보상받는 과정까지의 플로우 차트를 그린 것입니다. 플레이어는 코인 보상을 획득하여 상자에서 연료를 얻을 수 있습니다.

이와 같은 방식으로 플레이어는 연료, 덤벨을 추가적으로 획득하여 체중을 줄이고 체공시간을 늘려가며 잠긴 방 속에 숨겨진 열쇠를 찾아야 합니다.

메인 스테이지를 구상할 때 어려웠던 점은 맵과 캐릭터, 오브젝트들의 크기가 얼마나 크거나 작아야 때 적당한지가 가장 큰 난관이었습니다. 처음에는 맵의 크기를 가로 4개의 방, 세로 5개의 방으로 랜덤 생성을 통해 방의 성질을 생성하는 것을 고려하였으나 시간이 부족한 것은 물론이고 플레이 타임이 방대하게 늘어난다고 판단하여 가로 3개의방, 세로 3개의 방으로 변형시킨 대신 2층까지의 있는 것으로 구상하고 랜덤 생성을 포기하게 되어 아쉬움이 큽니다.


NPC에게 말을 걸어
첫 번째 퀘스트를 수주하기
Quest 1.
: 이동 & 점프 & 제트팩 사용
수행했는가?
No
대기
Yes
NPC에게 말을 걸어
두 번째 퀘스트를 수주하기
Quest 2.
: 덤벨 획득 & 체중감량
수행했는가?
No
대기
Yes
NPC에게 말을 걸어
세 번째 퀘스트를 수주하기
Quest 3.
: 빵 획득 & 체중증량
수행했는가?
No
대기
Yes
NPC에게 말을 걸어
무엇을 해야 하는지 듣기
다음 씬으로 자동 이동
1-9번 방의 덤벨 획득
덤벨을 사용하여
체중을 감량했는가?
No
자동 저장
Yes
2층으로 올라갔는가?
Yes
No
자동 저장
2-4번 시험의 방
: 무게저울 Lv1 도전
1-2번 시험의 방
: 패널퍼즐 Lv1 도전
성공했는가?
No
게임 오버
Yes
50코인 획득
...

구현 내용 및 결과

카메라

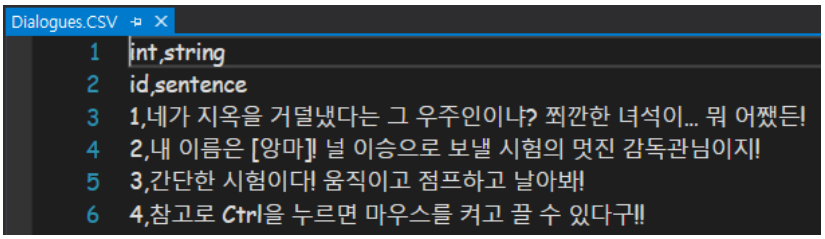
플레이어를 비추는 메인 카메라는 3인칭으로 구현했습니다. 오브젝트와 벽이 많은 게임에서 일반 3인칭 카메라는 오브젝트 뒤에 숨은 플레이어를 비추지 못해 플레이에 어려움이 있었습니다. 따라서 플레이어와 카메라 사이에 오브젝트가 있을 경우 해당 오브젝트 앞으로 카메라를 움직여 플레이어를 비추도록 구현하였습니다.

Dialogue Script

튜토리얼 스테이지에서 NPC의 대사 출력을 위해 Dialogue를 엑셀파일로 작성하고 이를 불러와서 출력하는 과정을 구현했습니다. 여기서 csv파일을 string의 형태로 parsing하는 부분은 NPCDialogue 클래스에서 진행되며, 대화창을 띄우고 텍스트를 출력해주는 DialogueManager에게 parsing한 문장을 보내주는 역할을 합니다.

Parsing의 과정은 다음과 같습니다.

- NPC가 마우스 클릭 이벤트를 감지하면 현재 단계에 맞는 스크립트를 불러와 parsing하여 DialogueManager 클래스에 보내줍니다.
- DialogueManager클래스에서는 받은 스크립트가 이전과 같은 것인지 확인하고 이전과 다르다면 이전 대사의 마지막 id번호를 기록하고 대사를 저장합니다.
- DialogueManager클래스에서 저장된 대사를 행 단위로 출력합니다. 이 때, 출력과 관련된 연출로 텍스트를 타이핑 치듯이 글자를 하나씩 덧붙이는 느낌으로 구현해봤습니다.

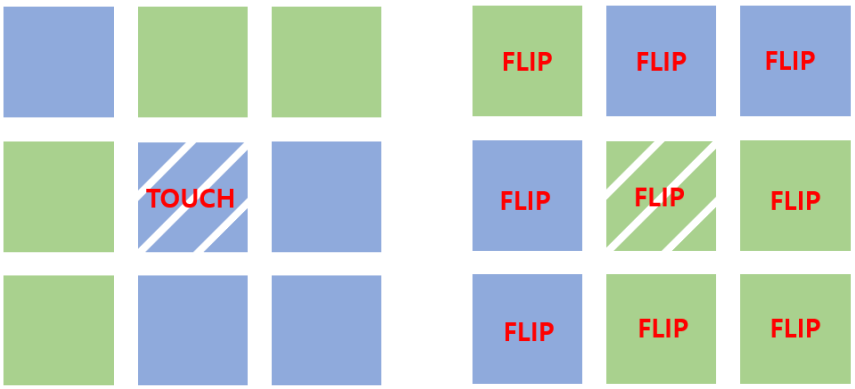


자동 저장

데이터 저장의 경우 json파일로 내보내고 필요할 때나 게임 중에 불러오기를 할 수 있도록 구현했습니다. Auto save기능으로 구현하여 플레이어가 특정 조건을 만족했을 때, 예를 들면 시험의 방에서 시험을 치룬 직후나 시험의 방에 입장할 때, 또는 게임을 종료할 때 마지막 위치에서 가장 가까운 세이브 포인트로 플레이어 위치를 저장하는 방법으로 고려했습니다.

패널 퍼즐

패널 퍼즐의 경우 2학년 2학기 객체지향프로그래밍(2)수업에서 구현했던 지뢰 찾기 게임을 응용하여 패널 퍼즐을 구현해봤습니다. 지뢰 찾기의 규칙 중 하나인 선택한 땅과 그 주위 여덟 방향의 땅이 모두 열리는 로직을 응용하여 게임에 적용한 결과로, 선택한 패널과 그 주위의 패널들이 함께 뒤집어지는 로직을 구현했습니다.



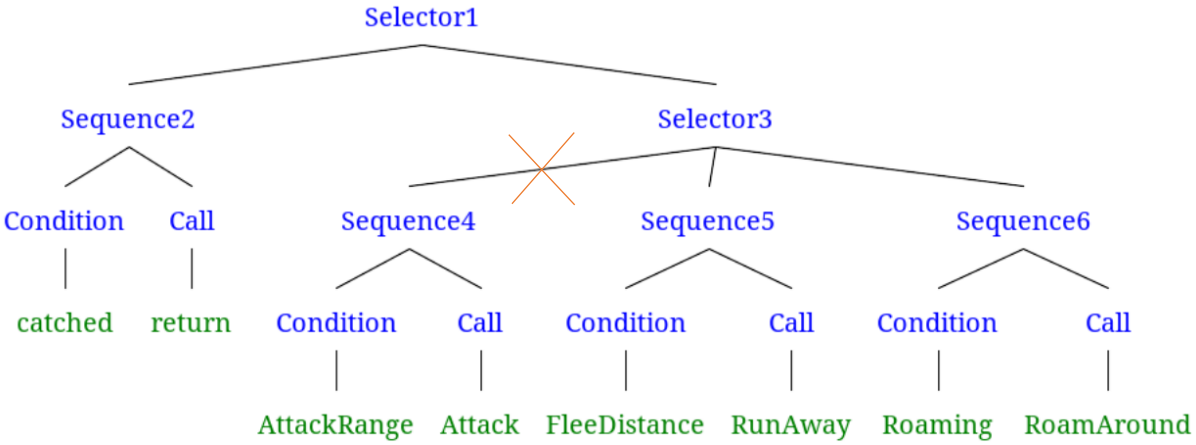
지뢰 찾기에서는 땅을 파서 지뢰가 있으면 게임오버가 되는 형식이지만 본 게임에서는 이를 조금 응용하여 해당 상태를 true 또는 false로 전환하여 왼쪽의 패널들과 같은 상태를 만들어야 하기 때문에 관련 부분을 게임에 맞게 수정했습니다.

기존 기획의 느낌인 패널의 상태를 변경한다는 컨셉을 살리면서도 뒤집힌 상태와 그렇지 않은 상태를 구분하기 위해 단순한 모양에 색을 입혀 차이를 주는 방법을 생각해봤습니다. (플레이를 소망하는 지인 중 색약이 있는 분이 있어 색약이 있는 플레이어도 쉽게 구분할 수 있도록 밝은 색과 어두운 색을 참고하여 차이를 주려

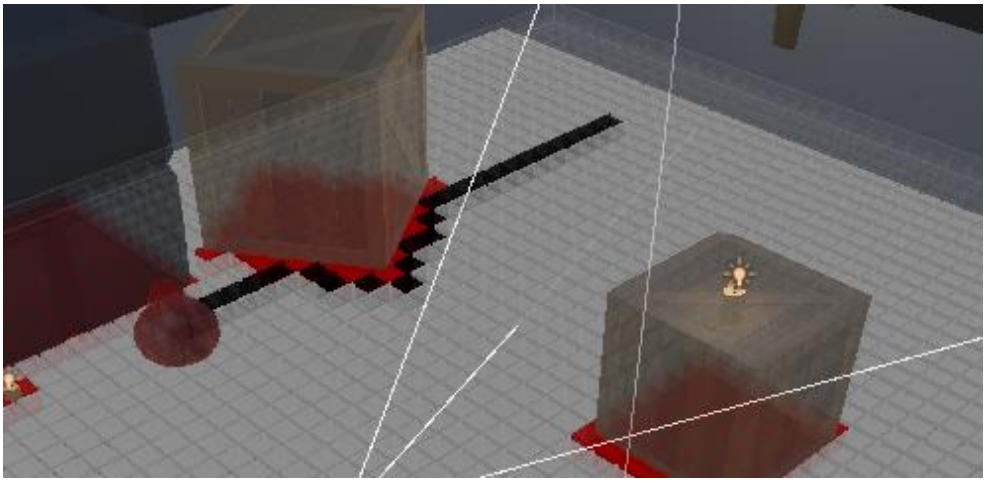
고 했습니다.)

몬스터

몬스터의 행동을 BT(Binary Tree)를 이용하여 계획했습니다. 아래의 그림은 구현 계획 당시의 BT입니다. 구현하면서 바뀐 점은 플레이어의 조작이 까다로워지는 것을 고려하여 공격(sequence4)관련 행동을 보류하게 됐습니다. 실제로 플레이어가 몬스터에게 다가가면 몬스터의 이동속도가 소폭으로 빨라지게 되는데, 공격을 하면 플레이어가 속수무책으로 당하게 되거나 스테이지를 클리어하기 어려워져서 잡힘 상태와 도망, 로밍 상태만 구현했습니다.



길 찾기 (A*알고리즘 사용)



2020년 3학년 2학기 알고리즘 강의에서 다뤘던 A*알고리즘을 이용하여 몬스터의

길 찾기를 구현했습니다. 해당 화면은 디버깅(Editor의 Scene View)에서만 보이는 화면이며, 붉은색으로 표시된 장애물은 피해 검은색 경로를 따라 이동하도록 구현했습니다.

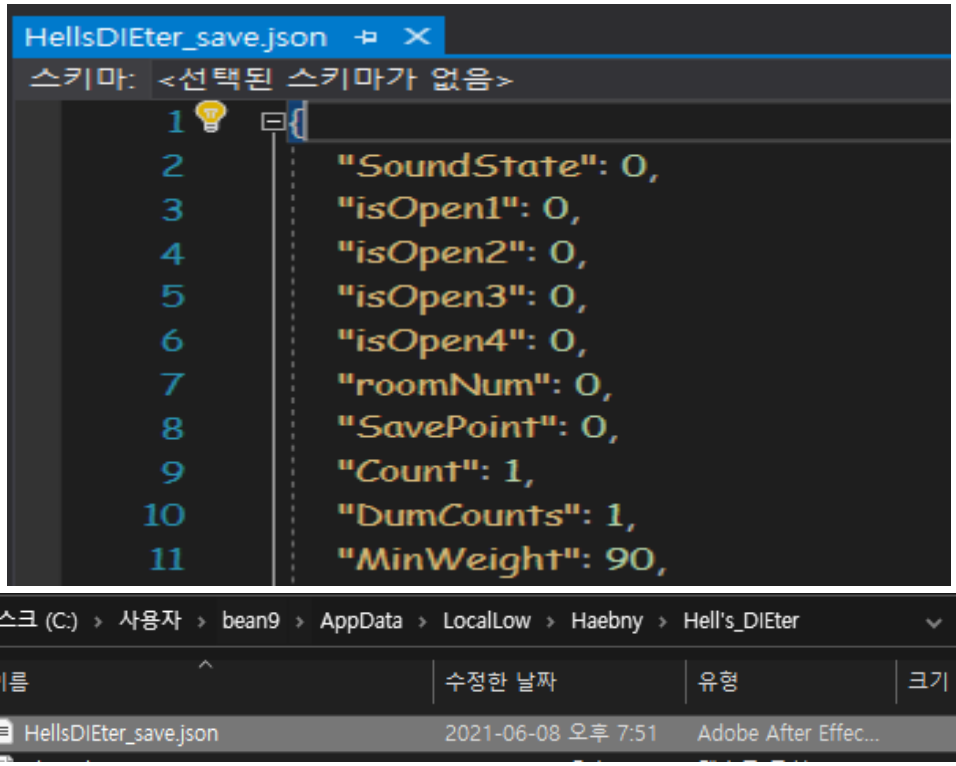
구현 시 몇 가지 어려웠던 부분이 있었는데, 게임에서 얼마나 가시적으로 보일지를 고민했으나 생각보다 드러나지 않아 좀 아쉬웠습니다. 또한 이미 배웠던 것이라 쉬울 것이라 생각했으나 제가 알고 있던 내용이 수박 겉핥기 식의 내용이었음을 알게 되고서는 개념 공부를 더 하느라 시간이 좀 오래 걸렸던 것 같습니다. 구현 중에 다른 알고리즘으로 바꿀까 고민도 많이 했는데, 포기하지 않고 A* 알고리즘을 사용하여 구현했기 때문에 해당 알고리즘의 이해가 더 깊어짐을 느낄 수 있었습니다.

비동기 씬 전환

메인 스테이지의 경우 다른 스테이지보다 오브젝트가 많은 편이기 때문에 장면을 이동할 때 화면이 멈추는 작업처리 지연 현상이 있었습니다. 이 문제를 해결하기 위하여 SceneLoader 클래스를 구현하여 Scene이 로드되는 중에는 로딩화면과 TIP을 제공하는 방식을 이용하여 구현해봤습니다. 로딩화면과 함께 출력되는 TIP의 경우 총 5가지가 있으며, 게임을 진행할 때 도움이 될 수 있는 텍스트를 위주로 출력합니다. 출력되는 순서는 랜덤이며, 큐(Queue)에 저장된 TIP들이 모두 Dequeue된다면 다시 무작위 순서로 TIP을 채우도록 구현했습니다.

비동기 씬 전환 방법은 유니티에서 씬 전환 방법을 고민하면서 조금 헤맨 부분이었습니다. 처음에는 다중 Scene을 로드하고 기존의 Scene을 Unload하여 새로 시험의 방 스테이지 Scene이 활성화되도록 하는 방법을 생각하였는데, UI간의 충돌이나 데이터 저장의 문제도 있었습니다. 이 과정에서 처음 시도한 데이터 저장 방식은 PlayerPrefs를 이용한 저장 방식이었습니다. 다만, PlayerPrefs는 많은 데이터를 저장할 때 적합하지 않을 뿐더러, 데이터 보호수준도 좋지 않다는 것을 알게 되었습니다. 데이터를 더욱 원활히 저장하고 불러오기 위해 Json으로 저장하고 불러오는 방식까지 발전시켰습니다. 첨부한 사진은 로컬에 저장된 데이터의 일부를 보여주며, 파일형식은 json으로 저장되었음을 나타내기 위해 첨부하였습니

다.



소감

이전 기획을 갈아 엎고 새로 시작한 기획으로 제작했기 때문에 3개월가량의 짧은 시간동안 구현하게 돼서 부족한 점과 구현하지못한 것들이 많다고 생각이 들어 아쉬운 마음이 가장 큽니다. 그렇지만 이번 졸업작품을 제작하면서 기획자, 프로그래머, 협업활동을 포함한 그래픽디자이너의 역할들을 하나씩 해볼 수 있었던 점이 저에게 큰 의미가 되었습니다.

앞으로의 향후 계획으로는 2D 그래픽 요소(UI, Title Scene Background Image 등) 작업에 있어 협업을 진행한 게임그래픽디자인전공 이동건 학우가 전역하기 전까지 게임을 좀 더 보완하여 PC버전을 완성하고 이를 안드로이드, IOS에서도 구동될 수 있도록 변환하는 작업을 진행하여 Google Play와 AppStore에 등록하는 것을 목표로 진행할 예정입니다. 또한, 구직 활동을 본격적으로 시작할 예정입니다.

더 열심히 더 잘하는 프로그래머가 될 수 있도록 항상 노력하고 정진하는 정해빈이 될 수 있도록 앞으로도 꾸준히 공부하고 도전해보도록 하겠습니다.