**게임제작프로젝트 제작 보고서**

**2019.11.12**

**게임소프트웨어 4학년**

**B477044 황성배**

**홍익대학교**

목 차

[1. 타이틀 2](#_Toc400822075)

[1.1. 싱글톤패턴의 씬관리 스크립트 제작 2](#_Toc400822076)

[1.2. 로딩 화면 구현 2](#_Toc400822077)

[1.3. 비디오 플레이어 구현 2](#_Toc400822077)

[2. 튜토리얼 제작 2](#_Toc400822078)

[3. 메인 게임씬 2](#_Toc400822084)

[4. 사운드 구현 3](#_Toc400822084)

[4.1. 사운드매니저 객체를 통한 배경음과 효과음을 통합관리 3](#_Toc400822076)

[4.2. 3D 사운드 효과 4](#_Toc400822076)

[5. 사용자 인터페이스 제작 4](#_Toc400822084)

[5.1. 기본 크로스헤어 애니메이션 제작 4](#_Toc400822076)

[5.2. 플레이어 스테이터스 구현 4](#_Toc400822076)

[5.3. 인벤토리 구현 4](#_Toc400822076)

[5.4. 일시정지 구현 4](#_Toc400822076)

[5.5. 미니맵 구현 5](#_Toc400822076)

[5.6. 건축 탭 구현(프리뷰 추가) 5](#_Toc400822076)

[5.7. 페이드인아웃 구현 5](#_Toc400822076)

[5.8. 대사창 구현 5](#_Toc400822076)

[5.9. 밤낮 효과 구현 5](#_Toc400822076)

[5.10. 획득가능 아이템 표시 6](#_Toc400822076)

[5.11. 버튼 마우스오버 애니메이션 효과적용 6](#_Toc400822076)

[5.12. 웨이포인트 이정표 제작 6](#_Toc400822076)

[6. 플레이어 근접무기 종류에따른 여러가지 다른 상호작용 구현 6](#_Toc400822084)

[7. 다양한 행동을 수행하는 NPC 구현(FSM) 7](#_Toc400822084)

[8. 세이브 로드 구현 8](#_Toc400822084)

[9. 느낀점 8](#_Toc400822084)

**1. 타이틀**

**1.1 싱글톤패턴의 씬관리 스크립트 제작**

타이틀에서부터 씬을 이동할 수 있으며 한번생성후 파괴되지 않는 싱글톤 패턴의 씬관리 스크립트를 제작하여 전체적인 씬 이동을 구현하였습니다.

**1.2 로딩 화면 구현**

씬이동시 로딩진행상황을 실시간으로 보여주기위해 유니티내AsyncOperation을 이용, 비동기적 씬로딩 코루틴함수를 작성하여 로딩화면을 구현했습니다.. 그후 유니티 내 지원하는 함수인 OnLevelWasLoaded()함수안에 씬이 완전히 로드되기전 Awake()함수보다 먼저 실행되야되는 코드(세이브된 데이터를 로드하는 작업 등)를 작성하여 타이틀간 이동시 먼저 수행해야 될 작업코드를 작성하였습니다.

**1.3 비디오 플레이어로 스토리 씬 및 타이틀 재생**

유니티 비디오 플레이어코드를 타이틀 씬 관리 스크립트에 추가하여 타이틀에서 배경영상과 스토리영상 재생을 구현했습니다.

**2. 튜토리얼 제작**

메인게임씬과 비슷한 환경에서 기본적인 게임방법을 설명해주는 튜토리얼 씬을 따로 만든 후 타이틀에서 선택하여 들어갈 수 있도록 하였습니다. 튜토리얼이 종료되면 자동으로 타이틀로 돌아가도록 구현했습니다.

**3. 메인 게임씬**

메인 게임 씬내에서 레벨디자인을 적용하여 낮(플레이어의 기본적인생존)스테이지와 좀더 난이도가 높은 밤(몬스터의 습격)스테이지를 번갈아 버티며 엔딩조건을 달성하는 방식으로 메인게임스테이지를 구현하였습니다.

**4. 사운드 구현**

**4.1 사운드 매니저객체를 통한 배경음과 효과음을 통합관리**

씬 전환간에도 파괴되지 않는 코드를 붙인 싱글톤 패턴의 스크립트로 작성하였고, 함수의 매개변수를 재생할 사운드의 이름을 받는 방식으로 통일하여 관리하였습니다. 스크립트내 함수에서는 사운드 재생 및 정지 할 수 있는 함수를 배경음과 효과음을 각각 구분하여 작성하였습니다.

**4.2 3D사운드 효과**

유니티내 기능인 3D Sound효과를 이용하여 플레이어가 소리가 나는 근원지 근처에 가까워질수록 소리가 더크게 들리는 효과를 적용하였습니다. Ex) 구조헬기소리가 멀리서는 안들리지만 플레이어가 헬기에 가까이 도착할수록 헬기 소리가 들리도록 했습니다.

**4.3 사운드 옵션 기능 추가**

타이틀내 환경설정 버튼에서 게임의 사운드를 조절할 수 있는 옵션을 추가하여 플레이어가 사운드 볼륨을 조절할 수 있도록 했습니다.

**5. 사용자 인터페이스 (UI) 적용**

**5.1 기본 크로스헤어 애니메이션 제작**

유니티 내 애니메이터을 통하여 크로스헤어 애니메이션을 프레임을 조절하여 직접 제작하여 달리기, 걷기, 앉기에 따라 크로스헤어의 움직임 강도를 조절하였습니다.

**5-2 게임 플레이에 필요한 플레이어의 스테이터스 관리 ( 배고픔, 목마름, 스테미너, 체력)**

플레이어의 상태수치를 여러 개 추가하여 스크립트를 통하여 관리했습니다. 목마름,배고픔 게이지를 추가한 후 점점 감소하도록 하였고 게임내에서 상호작용을 통해 음식을 먹으면 회복되도록 구현했습니다. 스테미나수치가 차있는경우 플레이어가 달릴수있도록 했고, 스테미너가 떨어지면 일정딜레이 시간을 체크한후 다시 회복하는 코드를 추가하여 스테미너를 관리했습니다.

**5-3 인벤토리 구현**

플레이어가 아이템을 습득하기위해 RayCast를 이용하여 얻은 아이템 정보를 받아온 후 아이템획득시 그 정보를 인벤토리에 저장하는 방식으로 구현했습니다.

유니티내 이벤트 시스템을 이용하여 인벤토리내 드래그앤 드롭 및 클릭을 했을시 아이템과 상호작용(ex: 장비아이템의 경우 장착, 소모품의 경우 사용)을 할 수 있도록 함수를 작성했습니다.

**5-4 일시정지 구현**

일시정지 버튼을 누르면 게임내 상황이 멈춘 상태로 메뉴에서 저장 및 불러오기 및 게임종료를 할 수 있도록 구성했습니다. 또한 일시정지 상태일때는 플레이어가 아무것도 하지 못하도록 bool 변수를 통해 제어했습니다.

**5-5 미니맵 구현**

플레이어가 대략 어디쯤에 있는지 위치를 파악하기위해 탑뷰방식으로 카메라를 하나 추가하여 미니맵을 작성했습니다.

**5-6 건축 탭 구현 (프리뷰추가)**

인벤토리 내 건축을 할 수 있는 재료아이템의 개수를 체크하는 함수를 작성하여 개수를 체크 후 아이템을 만들 수 있도록 하는 함수를 만들었습니다. 건축을 할때는 바로 건설하는게 아닌 일단 프리뷰오브젝트(연두색)를 생성하도록 했고 프리뷰오브젝트에 충돌박스를 체크하여 알맞은 위치에 있고, 재료가 충분한가의 조건을 동시에 만족하면 건축을 할 수 있도록했습니다. 이때 건축을 하고난 후 사용된 재료를 소모시키는 함수도 작성했습니다.

**5-7 페이드인아웃효과**

이벤트 중간중간 장면연출을 위해 페이드인 아웃 효과를 만들었습니다. 페이드인 아웃은 UI화면 전체에 이미지의 명암을 코루틴함수로 조절하여 구현했습니다.

**5-8 대사(JSON 라이브러리사용)**

제 게임은 대사량이 어느정도 있어서 관리를 쉽게하기위해 대사를 JSON으로 관리했습니다. 문자열을 JSON파일로 저장했고 불러올 때 코루틴을 사용하여 텍스트 애니메이션을 이용해

**5-9 밤낮 시계 UI**

디렉셔널라이팅의 회전값과 유니티내에서 지원하는 fog의 밀도를 조절하는 함수를 작성하여 밤낮효과를 만들었습니다.

**5-10 획득가능 아이템 표시(Raycast사용)**

플레이어가 상시 Raycast를이용하여 눈앞에 획득가능한 아이템이 존재할시 화면에 아이템이 획득 가능하다는 정보를 띄우는 부분을 작성했습니다.

**5-11 각 버튼 애니메이션 효과적용(마우스 오버시)**

버튼에 커서를 가져다대면 상호작용으로 커졌다 작아졌다 하는 애니메이션을 추가하여 좀더 UI를 화려하게 꾸몄습니다.

**5-12 WayPoint Arrow시스템 제작**

특정 이벤트 진행 후 플레이어가 길을 쉽게 찾을 수 있도록 하는 WayPoint Arrow를 제작했습니다. 이 화살표는 생성된 이후 플레이어 근처에 계속 있으며 목적지까지 방향을 제시합니다. 이후 목적지에 도달하면 사라지게 되고 해당하는 이벤트를 진행하도록 작성했습니다.

**6. 플레이어 근접무기 종류에따른 다른 효과 구현**

플레이어가 손에 들고있는 도구의 상태에 따라 게임내 다양한 여러 오브젝트에 상호작용을 할 수 있는가 여부를 구분할 수 있게 했습니다. 기본적인 기능을 가지고있는 스크립트를 만든 후 상속하여 세분화 된 각각의 도구 스크립트를 제작했습니다.

**7. 다양한 행동을 수행하는 NPC 구현(FSM)**

게임내 존재하는 NPC는 공격형과 비공격형의 두종류의 NPC로 제작했습니다. 기본적으로 NPC는 일정시간마다 랜덤한 패턴으로 정해진 동작을 수행합니다.( ex: 두리번거리기, 풀뜯기, 걷기, 잠시 가만히있기)

이러한 동작을 수행하다 플레이어가 NPC시야각 안에 들어오게 되면 비공격형 NPC는 플레이어의 반대방향으로 도망을 가고, 공격형 NPC는 유니티내 NavMesh기능을 활용해 플레이어를 뒤쫒은 후 공격모션으로 들어가도록 설계했습니다.

**8. 세이브 로드 구현(JSON형식)**

유저의 세이브데이터를 JSON형식으로 변환하여 저장 및 불러오기를 구현했습니다. JSON형식으로 저장할 때 유니티에서 지원하는 자료형의 형태가 저장이 불가능하기 때문에 각종 저장데이터 정보를 int, string등의 형태로 변환하여 저장했습니다. 이러한 과정이 조금 어렵게 느껴졌습니다.