2017년 여름합숙

프로젝트 최종보고서

<**졸업해주세요 공대생이여! (For 17학번)>**

**–학점 키우기**

제출일: 2017년 07월 28일

|  |  |
| --- | --- |
| 기수 | 31 |
| 이름 | 임에딘 |

1. 프로젝트 개요

* 1. **주제 선정 배경**

4주간의 프로젝트로 주제 선정에 있어서 1학년 1학기동안 배웠던 C++에서 더 나아가 C#을 스크립트 언어로 사용이 가능한 유니티3D 게임을 제작해보고 싶었다. 결과가 어떠한지 눈으로 바로 확인할 수 있고 디자인 면에서 혼자 원하는대로 구상할 수 있으며, 또한 게임을 구현하는 과정에서 지속적으로 재미를 느낄 수 있을 것 같은 이유가 주제 선정에 있어서 가장 컸다. 한 학기동안 과제에 시달리며 삶의 여유를 느끼지 못한 것을 되돌아보며 게임 캐릭터를 17학번의 공대생으로 두었고, 해커동아리 31기 동기들이 아무탈 없이 무사히 졸업을 하길 바라는 마음으로 게임을 제작하기로 했다.

* 1. **문제 정의**

유니티 엔진에서 제공하는 여러 기능들을 쓰는 방법 익히기, C# 스크립트에서 어떤 함수들을 써야 원하는 방식으로 게임을 구현할 수 있는지.

* 1. **기대 효과**

게임을 즐기는 입장에서 벗어나 직접 구현해보는 입장이 된다. 객체지향인 C#을 통해 여러 스크립트들을 처음으로 동시에 써보는 기회가 된다. 플레이어는 학점의 중요성을 조금은 깨닫는다.

* 1. **프로젝트 요구 사항**

크게 2가지로 나눌 수 있는데 유니티 엔진에서 제공하는 여러 기능들을 쓰는 방법, 그리고 스크립트 작성에 있어서 유니티의 모든 스크립트가 갖는 MonoBehaviour이란 기본 클래스 내의 여러 함수들과 변수들을 쓰는 방법

* 1. **최초 기획 및 변경 사항**

『3D필드 위에서 공대생이 학점을 채워나가며 게임 클리어하기』

최초 기획한 것에서 크게 변경된 사항은 없으며 7월 3일날 발표했을 때 선배님들의 피드백을 받아 변경한 것과 도중에 변경한 사항은 다음과 같다 :

* 1,2,3,4학년 1,2학기로 총 8개의 스테이지를 만들기에 너무 많지 않은가? -> 총 4개의 스테이지로 변경
* 점수를 올려주는 기능에 대해 -> 4개의 스테이지마다 다른 월드 좌표로 x,y,z축을 갖고 있는데 그 위에 캐릭터나 장애물들을 배치하려면 좌표를 일일이 설정해야 한다. ‘학점’아이콘들과 포털, 캐릭터, 적들 등을 설정하는데에도 시간이 많이 걸렸으며 점수를 올려주는 아이템까지 놓기에는 다른 것들을 구현하는 것이 더 우선순위가 되어 시간이 너무 촉박했다.
* 이중 점프나 가속, 엎드려 가기 등의 행위는 캐릭터 컨트롤러 컴포넌트를 사용하면 가능하다. 하지만 그것보다 RigidBody 컴포넌트와, Capsule Collider 컴포넌트를 활용하는 것이 스크립트를 통해 움직임을 더 제어할 수 있음을 후에 깨닫고 2주차 초반에 바꾸었고 주인공이 적들을 피해가며 도망가는 것을 주 이동로직으로 설정하였다.

2. 프로젝트 내용

* 1. **개발 언어**

C# : 유니티 엔진은 C#과 자바스크립트를 지원하는데 후자는 실질적으로는 그의 문법적인 요소만 차용한 스크립트 언어로 실제와는 조금 다르다고 한다. 또한 C++와 비슷한 연산자를 가진 C#을 골라서 게임내용에 조금이라도 더 치중할 수 있도록 하였다.

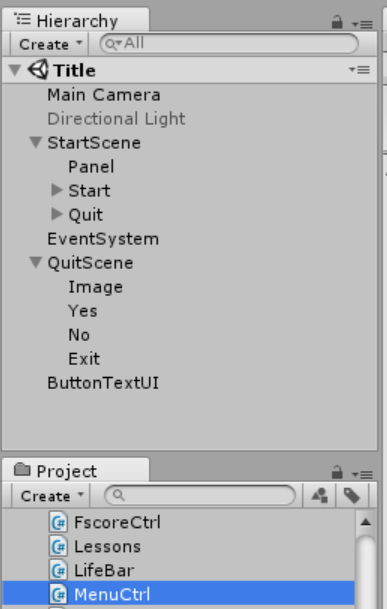
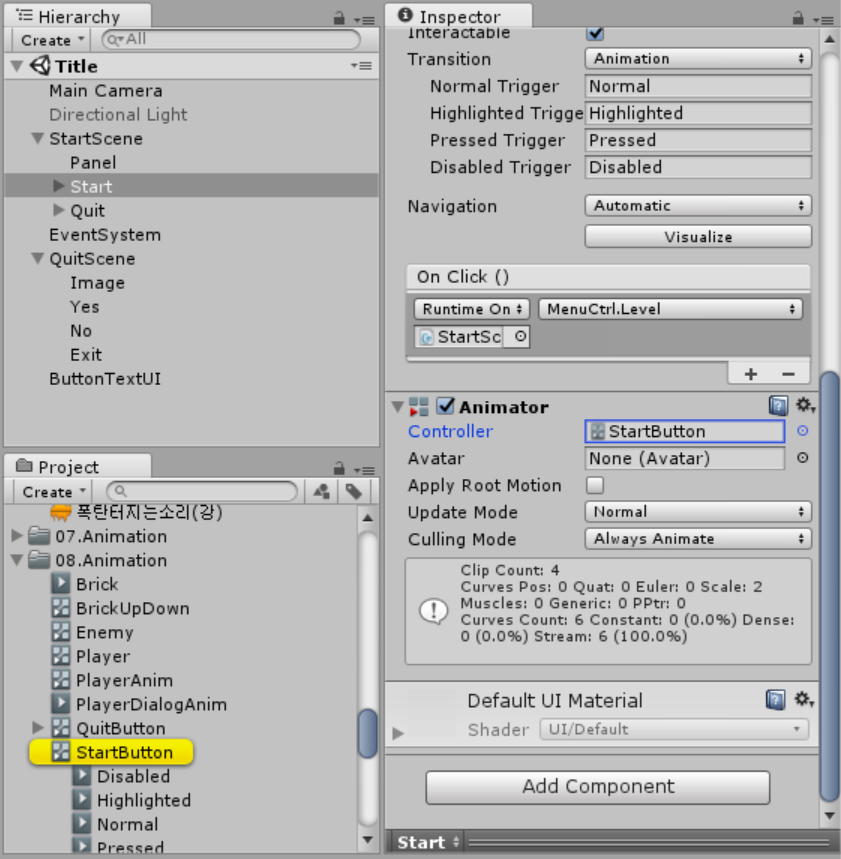
* 1. **개발 도구**

유니티 엔진 : 게임 엔진들 중 가장 유명한 2가지 중 하나로 게임을 다른 엔진들에 비해 비교적 쉽게 구현할 수 있다고 한다.

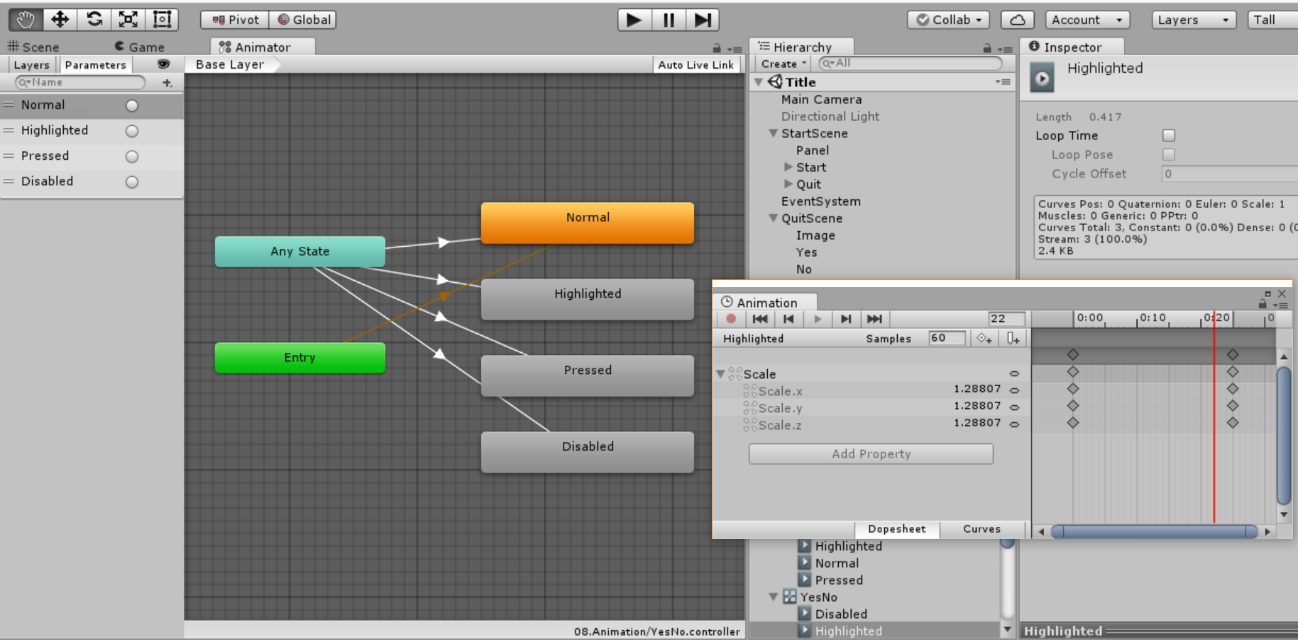
* 1. **시나리오 흐름도**
     1. **전체 시나리오 흐름도**
* 총 4가지 스테이지가 있으며 build를 통해 지정된 Scene들의 번호를 a학기와 b학기 할 때 a학기에 값을 넣어지고 게임이 시작된다.
* 첫 번째 Scene : 시작화면을 띄우면 Start버튼과 Quit버튼이 있다. 필요한 게임 오브젝트들은 ‘StartScene’의 Canvas에 Panel(이미지 동시에 넣음)과 Start버튼과 Quit버튼, 그리고 ‘QuitScene’의 Canvas와 이미지, Yes&No&Exit 버튼을 만들었다. 이 씬을 제어하는 스크립트는 MenuCtrl로, 각 버튼을 눌렀을 때 버튼 컴포넌트의 OnClick 기능을 이용해서 스크립트 내의 함수를 호출할 수 있다. 또한 애니메이션 컨트롤러를 만들어서 메카님 애니메이션을 이용해 마우스를 버튼 위로 갖다댈 때 State가 Highlighted로 바뀌면서 예를 들어 Quit화면의 Yes/No 버튼의 경우에는 Scale을 1에서 1.3으로 바뀌게 했다. void Start()함수에서 ‘QuitScene’캔버스를 false로 두어 보이지 않게 한다.

◎Start버튼→Using UnityEngine.SceneManagement;를 쓰고 public void Level(){ SceneManager.LoadScene(5); }라는 함수를 써서 OnClick에 Level()함수를 연결하면 Build시 지정된 5번째 씬으로 넘어가게 된다. (이 때 5번째는 게임 시작할 때 튜토리얼로 대화창이 나오면서 캐릭터가 설명하는 씬이지만 a학기와 b학기의 계산 용이함을 위해 0,1,2,~순으로 가지 않고 5번째로 두어 1은 a=1(1학년), 2는 a=2(2학년), 이런 식으로 갈 수 있게 했다.)

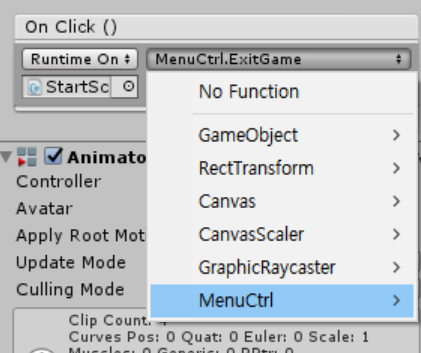
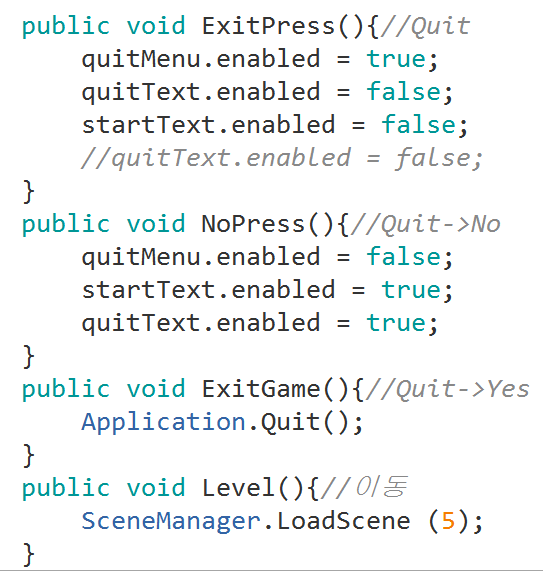
◎Quit버튼→함수를 만들어 Quit캔버스를 띄울 수 있도록 quitMenu.enabled=true;라고 하고 동시에 앞에 있었던 버튼 2개를 작동하지 않게 버튼들을 false로 한다. Quit캔버스가 보이면 ‘게임을 종료하시겠습니까?’와 함께 Yes와 No 2가지 버튼이 나온다. (추가적으로 오른쪽 위의 X표도 만들었는데 No와 같다)

∆시작 씬에 필요한 것들 ∆버튼에 애니메이터 컨트롤러가 있다



∆애니메이터 컨트롤러의 Highlighted 애니메이션에서 Ctrl+6으로 애니메이션창을 띄워서 Scale의 x,y,z축을 프레임따라 어떻게 바꿀지 설정한다

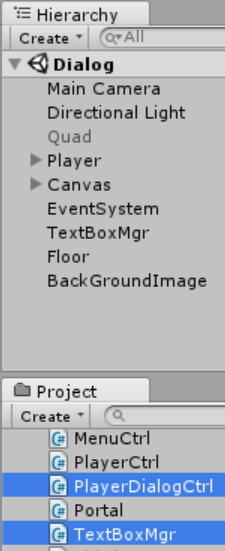
 

∆버튼 클릭할 때 발생하는 함수 설정 ∆버튼을 누르면 발생할 함수들 설정

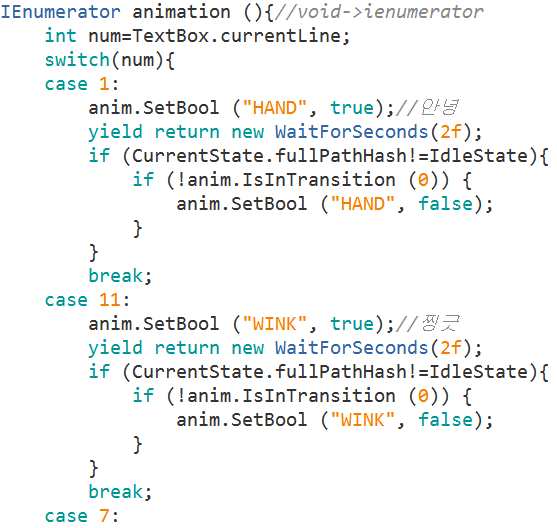
* 두 번째 Scene : 캐릭터가 게임의 대략적인 내용과 컨트롤 방법을 설명해주는 대화창을 구현한 것이다. 보통 대화창을 구현함에 있어서 itween이라는 유니티의 부가적인 플러그인을 사용하는 경우가 많았다. 하지만 이 캐릭터의 2D모습을 갖고 있지 않기도 하고, 대화의 문장 번호를 받아 switch문에서 그에 따라 캐릭터에게 애니메이터 컨트롤러에서 상황에 따른 메카님 애니메이션을 이용해 구현하고 싶었다. 이 방법은 혼자 고안해냈다! 우선 capsule collider을 갖고 있는 캐릭터가 중력을 받고 아래로 떨어지지 않도록 mesh collider을 갖는 Plane이라는 게임 오브젝트를 생성하고 이름을 ‘Floor’로 바꾼다. 카메라의 위치는 화면이 카메라에 다 들어오도록 설정하고 고정시킨다.

◎우선 TextBoxMgr은 문장들을 써놓은 text파일을 임포트한걸 불러와 엔터 친 것을 단위로 나눠서 string[] textLines에 넣게 했다. 캐릭터가 말할 줄의 번호는 int currentLine이라는 함수를 썼다. 매 프레임마다 호출하는 Update()함수에서, 만약 엔터키로 입력값을 받으면 PlayerDialogCtrl 스크립트에 있는 코루틴을 실행시키며(애니메이션 관련) 동시에 currentLine++를 한다. 만약 currentLine이 문장 배열의 마지막까지 갔을 때 대화창을 안보이게 하며 동시에 게임을 실행할 수 있는 씬으로 넘어가게 하는 함수인 DisableTextBox()를 호출한다. 이 스크립트에는 코루틴함수가 하나 더있는데, 바로 문장을 받았을 때 foreach를 통해 문장 내의 글자들을 배열로 넣게 되어 화면에 띄워지는 Text가 글자 하나하나를 띄우되 그 간격을 yield return null을 사용하여 한 프레임이 지나갔을 때 실행하도록 한다. 물론 yield return new WaitForSeconds(float time)을 쓰는 것도 나쁘지 않다! 프레임마다 호출하게 되서 평소에 RPG게임의 ‘옆으로 지나가는’ 대화창 같아보인다.

◎PlayerDialogCtrl 스크립트는 코루틴 함수와 switch문을 이용하여 캐릭터가 대화에 따른 애니메이션 상태를 변화시켜준다. 코루틴으로 만든 이유는 메카님 애니메이션 간의 Transition에서 Has Exit Time을 체크한 상태인데도 (다음 애니메이션이 실행되어야 해도 끝까지 한 다음 넘어가게 하는 기능) 바로 다음의 if문에서 가만히 있는 애니메이션이 아닌 상태일때(if첫번째) 그 애니메이션이 끝나야! (if두번째) 그 애니메이션을 false로 해야하지만 Debug를 해보니 true와 false를 계속 반복했다. 그래서 yield를 넣기로 했고 성공적이었다. TextBoxCtrl 스크립트에서 PlayerDialogCtrl의 코루틴을 실행시키고자 할 때 인터넷을 서치해봐도 찾은 방법들은 유니티5.6과 맞지 않았다. 그래서 생각한 것은 이 PlayerDialogCtrl에 void함수를 만들어서 그 내부에서 StartCoroutine를 썼다.

∆필요한 오브젝트와 스크립트 ∆3D 애니메이션을 위해 선택한 방식



∆ 코루틴 함수 내 switch문의 일부

* 1, 2, 3, 4 번째 스테이지들

게임 오브젝트들이 대부분 겹치며 월드 좌표계 x,y,z축과 스케일변경을 하는 과정을 거치면 된다. 게임 오브젝트들은 다음과 같다 : Main Camera, Player, Enemy(+Eyes), Terrain(또는 Plane), 일시정지 버튼을 누르면 나타나는 Canvas, 다음 스테이지로 넘어갈 때 나타나는 Canvas, 그 캔버스 아래로 나타나는 여러 버튼들과 OnClick을 통해 함수를 호출할 수 있게 하는 EventSystem 오브젝트, 게임에서 a학년 b학기와 플레이어의 상태 그리고 멘탈게이지의 증감을 볼 수 있는 Canvas 그리고 필드 위의 정적인 물체들이 있다.

◎Main Camera의 경우 FollowCam 스크립트를 통해 선형보간을 하여 플레이어의 Transform을 타겟으로 설정하고 플레이어로부터의 뒤로 distance와 위로 height을 이용해 따라다니도록 한다. ◎Player의 컨트롤은 PlayerCtrl 스크립트에서 전부 제어할 수 있도록 하였다.

◎포털에 들어갈 때 void OnTriggerEnter을 통해 어떠한 collider을 컴포넌트로 갖고 있는 게임 오브젝트가 들어가면 실행되는 함수를 이용하였다.

◎그 외에도 폭탄을 설치해서 사정범위안에 들어가면 3초동안 카운트다운하는 효과음을 내게 한 후 코루틴 함수를 이용해서 3초 뒤에 폭발효과를 나타내는 Prefab를 동적생성한 후, 동시에 그 폭탄 오브젝트를 Destroy한다. 캐릭터가 범위 안에 들어가면 카운트다운이 시작되는데, 카운트다운했는데도 여전히 범위안에 있으면 비명효과음과 함께 멘탈게이지를 덜어뜨렸다. 이것을 제어하는 스크립트는 BombCtrl이다.

◎멘탈 게이지바를 관리하는 스크립트는 LifeBar.cs이며, Canvas내의 LifeBar 오브젝트의 이미지에서 filled로 설정하여 horizontal로 왼쪽으로 감소하게 만들고 이를 이용해서 hp를 총괄하였다.

◎CoinSpawner은 맵이 비교적 간단한 2번째 스테이지에서 랜덤으로 x,y(고정),z축에 범위를 설정하여 for문으로 7번 돌리게 하였다. 나머지 스테이지들은 Prefab의 colone을 미리 배치해놓았다. 스테이지들의 하늘은 Window의 Lighting에서 Skybox에서 원하는 배경들을 넣었는데 대부분 어둡게 설정해서 (태양 같은 광원인 directional light를 게임 맵의 분위기 상 전부 해제시켰다) 코인들 자체에서도 point light을 통해 빛을 발산하게 하고 코인을 먹은 경우 Destroy가 되어 그 자식인 pointlight까지 사라진다.

◎Navigation에서 NavMesh를 Bake함으로서 Nav Mesh Agent를 갖고 있는 게임 오브젝트(적)가 돌아다닐 수 있는 범위를 설정한다. 적의 이동로직은 유니티에서 제공하고 있는 using UnityEngine.AI와 Nav Mesh Agent 컴포넌트를 이용해 Enemy.cs스크립트에서 제어를 했다.

* + 1. 모듈(기능)별 시나리오 흐름도 (일부 자세한 것만)
* 적 구현 (FscoreCtrl)

적의 상태들을 선언해서 Nav Mesh Agent를 통해 Listen, Move, Chase로 제어했다. 초기의 상태는 Listen으로 만약 그 상태의 경우 랜덤으로 적으로부터 15f만큼의 구형태에서 RandomPosition으로 위치를 받고 SamplePosition을 통해 현재의 위치에서 거기까지의 위치를 계산하며 그곳으로 향해 SetDestination을 이용해서 걸어가며 이때의 상태는 Move이다. 적에게 차일드화된 Eye 컴포넌트가 있는데 Cube로 기다란 직육면체를 만들어서 마치 시야처럼 만들었고 IsTrigger을 체크해서 만약 캐릭터가 그 범위안에 들어가면 캐릭터를 향해 속도롤 높이고 달려가게 만들었다. 이때의 상태는 Chase이고 그 Eye안으로 캐릭터가 들어갔는지에 대해서는 PlayerCtrl에서 코인과 함께 OnTriggerEnter함수로 제어했다.

* 1. 문제점 및 해결방법

수많은 문제점에 직면하고 해결해왔지만 그것들 중에서도 가장 특별했던 것을 크게 5가지로 나눠보았다.

1) Unity 5.6버전과 호환되지 않는 것들에 대해

1. 예를 들면 Application.LoadLevel(int num)이란 함수가 SceneManager.LoadScene(int num)으로 바뀌는 등 Unity 3,4, 심지어 5의 초반부 버젼에 비해 바뀐 것이 너무 많아서 인터넷에서 서치하고 썼더니 에러나서 다시 Unity5에 맞는 것으로 서치하는 방식이 잦았다. (애초에 Unity5버전으로 찾아봤어야지 라는 건 의미가 없다. 사람들이 유니티 공식사이트에 서로 질의응답해가며 ‘이러한 경우 이렇게 쓰면 돼요’ 식의 글이 많아서 이쪽으로 서치하게 되며 정보도 예전 버전의 경우가 많았다.)
2. 예전 버전에서는 사운드 파일들을 배열형태인 AudioClip[] \_audio;라고 하고 한 게임 오브젝트 내의 사운드 컴포넌트에서 여러 파일들을 넣은 다음 Element의 숫자대로 스크립트에서 콜하면 그에 해당되는 노래가 나왔으나 유니티 5.6은 이 기능이 더 이상 사용되지 않는다. AudioClip[]은 여전히 쓸 수 있지만 Inspector뷰에서 여러 개의 노래들을 넣지도 못하고 공식사이트에서도 어떻게 해야 활용할 수 있는지 description을 내놓지 않았다. 즉 그 게임 오브젝트 내부에 Audio Source 컴포넌트로 여러 개의 노래를 넣지 못한다. 그래서 누구도 고안하지 않은 나의 아이디어는 빈 게임 오브젝트를 만든 후 Projector뷰에 옮겨서 Prefab을 만든 후 거기에 Audio Source 컴포넌트를 집어넣은 후 A라는 스크립트에 B라는 함수가 발생하면 C라는 노래를 재생하고 싶을 때 Start함수에서 그 게임 오브젝트 내의 Audio Source컴포넌트에 접근해서 초기화하고 활용하는 것이었다.

2) NGUI는 95달러이다!

1. 유니티에서 한글 폰트 파일(ttf)을 집어넣으면 적용이 안됬다. 기본 폰트인 Arial을 쓰자니 마음이 아팠다. 구글링을 수도 없이 해봤지만 방법이 정말로 없었던 것 같았다. 계속 서치를 해보니 어떤 분이 NGUI를 다운받으면 쉽게 적용이 가능하다고 했다! 하지만 애셋스토어에서 검색해보니 95달러였다. (한국인이 NGUI없이 유니티를 강의할 때 Text컴포넌트에 아무도 한국어 폰트를 다운받아 쓴 사람이 없었다. 전부 영어였다. 1인 개발자라면 더욱 그러했다.) 반대로 여러 명이 개발했다면 디자인맡는 사람은 따로 원하는 글씨를 디자인하고 이미지화해서 배경을 투명화하여 적용했을 것이다. 이러한 생각을 해보니 (예를 들면 튜토리얼처럼) 문장 단위로 애니메이션(같은 착각을 일으키는 효과이지만 사실은 프레임마다 한 글자씩 반환받아 나타내는 것이다)까지 들어가며 나타내는 건 이미지화하는 것은 의미없는 노가다일 것이다. 따라서 적 캐릭터 위에 이름을 표시할 때와 ‘과제폭탄’이라는 게임 오브젝트 위의 Text같이 글씨가 적은 것을 이미지화하여 활용하기로 했다. 방법은 다음과 같이 했다 : 기존에 갖고 있던 클립 스튜디오를 이용해서 레이어에 원하는 글씨를 쓰고 jpeg파일로 저장한 다음 GitHub에서 WhiteEraser 프로그램을 다운받아서 배경을 투명화하고, 그 이미지 파일을 유니티에 임포트해서 Texture Type을 Defalt에서 Sprite(2D and UI)로 바꿨다.

3) 폰트는 어떻게 했어도 문제는 게임 오브젝트에 그것을 입히게 하는 것

1. 먼저 ‘과제폭탄’의 경우 GameObject 3D에서 이미지를 유일하게 입힐 수 있었던 Box Collider이 컴포넌트로 들어간 Cube를 선택했다. 렌더링모드는 CutOut(투명)으로 바꾸고(글씨만 보이게 하기 위함) Material에서 Albedo의 이미지를 설정한 후, 아래에 Emission을 체크하고 원하는 색깔로 설정하였다. 아쉽게도 정육각형 6면 전부 글씨가 나타나고 어떤 면만 글씨가 안보이게 할 수는 없는 것 같다.
2. ‘과제폭탄’에 이어서 두 번째로 적 위에 이름을 띄울 때에 관해서이다. 어느 유투브 영상에도 NGUI를 쓰지 않는 경우 적 위에 Text를 띄우지 않았다. 오직 라이프바만 띄웠다. 나중에 해봐서 알게 됬지만 Text가 블렌딩된 것 처럼 흐려지고 절대로 원하는대로 나오지 않았다. Image컴포넌트를 써서 위의 이미지화한 글씨체를 넣어도 ‘과제폭탄’처럼 글씨의 Emission기능이 없기 때문에 검은 색깔로 나온다. 그렇다면 Image가 아니라 Cube의 Material처럼 어떠한 게임 오브젝트에 적용할 수 있는 Material을 만들면 괜찮을 것 같았다. 그래서 마감일을 앞두고 생각해낸 나만의 방법은, Image 또는 Text 컴포넌트 대신에 보통 바닥에 쓰이는 Plane이란 게임 오브젝트를 만들어서 ‘적’ 게임오브젝트 아래로 차일드화(계층)한 후 Scale을 작게 한 후 가로세로의 비율을 대략 4:1로 하고 Material을 새로 만들어서 적용했다. Ctrl+D로 붙이기해서 하나는 rotation의 z값을 180도로 해서 게이머가 적의 앞뒤로 있을 때 글씨가 보이게 설정했다! 물론 캐릭터가 적을 공격할 수 있는 메카님 애니메이션과 함수들, 변수들을 설정했다면 라이프바도 넣었겠지만 기간이 짧은 것도 있었고 좀 더 긴장감을 위해 오직 달려가며 살아남아가는 게임 방식을 선택했다.

4) Window→Navigation에서 NavMesh를 Bake할 때 문제점

1. 다운 받은 필드들 중에서는, Terrain이 있다면 그 내부에 Fence같은 것들을 설치해서 그 안에서 플레이할 수 있도록 되어있었다. 하지만 NavMesh를 Bake(참고로 어떤 게임 오브젝트에 Nav Mesh Agent 컴포넌트를 추가하면 Bake된 것에 따라 이동경로가 정해진다.)하면 게임 오브젝트들 중 Navigation Static이 설정된 것들이 선택되는데 필드의 x,y,z좌표로 어디서부터 어디까지는 하지 말라고 설정할 수가 없었다. 그래서 그 지형은 static을 푼 다음 Plane을 적이 이동할 필드의 범위만큼 스케일을 바꿔서 navigation static을 설정하고 Bake했다. 이것도 내가 생각해낸 방법!

5) 에러들 (유니티 엔진 기능 내에서)

1. 동전들의 무한개 동적생성 : 분명 CoinSpawner 스크립트에서는 for문을 돌려서 7개만 돌리라고 했는데 2주차 금요일 8시반에 갑자기 유니티 게임을 실행할 때마다 에러가 났다. 스크립트 상의 문제가 아니었기에 콘솔창에 에러도 역시 나타나지 않아서 불안했었다. 우선 Build를 하고 실행해보니 필드 위에 동전들이 정말 무수히 많았다. 즉 동적 생성한 코인에서 Clone이 생기고 또 Clone이 생겨서 무한개로 생성된 것이었다. 코인과 관련된 것들을 전부 봐가며 여러가지 경우의 수로 시행착오를 해봤었는데 게임 Prefab에 들어있던 스크립트 컴포넌트를 잠시 해제시키니 바로 해결이 되었다. 게임오브젝트와 프리팹에 스크립트를 동시에 넣으면 안됨을 깨달았다.
2. 7월 25일 화요일날 게임을 build하는 과정에서 에러가 발생했다. 가장 초조했었던 것 같다. 7/11 bake visibility(4jobs)라며 진행중이라 하고 거기서 몇분동안 그대로인 것이었다. 그래서 찾아보니 어떤 사람은 3일이나 기다려도 마찬가지였다고 한다. 찾아보니 유니티의 (애셋들)스케일이 너무 크면 발생한다고 했다. 해결방법은 Window→Lighting에서 아래에 있는 Auto를 언체크함으로서 컴퓨터에서 자동으로 수시로 맵을 bake하지 않게 한다. 또한 lightmap 파라미터 object를 추가해서 1을 0.01로 낮춘 후 Lighting뷰에서 parameter을 이 오브젝트로 설정함으로서 유니티 엔진이 돌아가는데에 부담이 없도록 설정했다.

6) 에러들 (스크립트 내에서)

1. NullReferenceException 에러는 거의 매일 보는 것 같았다. public과 private문제라던가, private라도 분명 void Start()함수에서 변수를 ‘어떠한 게임 오브젝트의 이러한 컴포넌트’라고 선언과 동시에 초기화했는데 어쩌다가 에러날 때도 있다. 스크립트에서 벗어나 유니티 엔진 기능 자체에서도 어느 게임 오브젝트의 스크립트 내의 public 변수에 직접 오브젝트를 갖다놓지 않은 경우도 에러가 난다. 이의 경우 스테이지를 복수개를 만들 때 자주 발생했다.
2. 유니티에서 제공하는 MonoDevelop을 이용해서 스크립트를 짠다. 게임 실행할 때 콘솔창에 에러는 나지않았는데 뭔가 함수나 변수가 적용이 되지 않는 것 같다. 유니티를 쓸 때 대문자/소문자를 항상 주의해야 한다. 하나라도 틀리면 에러는 안나면서 게임이 이상하게 진행되버린다.

3.결론 및 후기

유니티 엔진으로 게임을 제작할 때 4주내내 흥미를 잃지 않고 재밌게 만들어갔던 것 같다. 여러가지 기능들이 제공되기 때문에 언제든지 무언가를 더 구현하고 싶으면 가지처럼 뻗어나가서 더 추가해나갈 수 있었고 공부하면 공부할수록 더 많은 것들을 구현해나갈 수 있었다. 또한 1학년 1학기동안은 한 스크립트 내에서 모든 것을 써내려갔지만 유니티를 공부하다보니 점점 스크립트들이 늘어나며 서로 스크립트간의 상호작용을 생각하게 되고 좀 더 다차원적으로 구현해 나가는 느낌을 많이 받았다. 어떠한 문제점에 직면했을 때 아무리 인터넷에 서치해도 해결방법이 나오지 않았을 때 나만의 방법으로 새로 고안하고 그것이 해결이 가능했을 때 쾌감이 가장 컸던 것 같았다. 특히 내 게임의 완성을 매일매일 노래하며 기다리던 친구에게 Demo판을 보여줘가면서 정말로 4주간 ‘완성’하고 싶은 마음이 강했고 주말알바를 한달 간 쉬기로 한 후 학교에서 11시간 동안 있을 때 점심시간이랑 저녁시간 외에는 하루종일 의자에 앉아서 게임 만드는 생각만 했다. Trello에서 일과표를 써가면서 예정대로라면 금요일날 점수 구현까지 완성하는 거였는데 4주 중에 2번이나 편두통이 찾아와서 하루종일 앓아버렸다. (약 먹고 진정됬지만 머리가 맹하다) 특히 3주차의 일요일날은 어디 갔다와서 저녁에 너무 피곤해서 밤8시에 쓰러졌었다. 다음 날 아침엔 눈물을 흩뿌리며 깬 것 같다. 이 시간들만 합쳤어도 게임을 완성시킬 수 있었을지 않았을까 하는 생각이 든다. 내가 기획한 게임의 스케일은 4주 내지 5주에 걸치면 완성이 가능한 것 같다. 중간에 가지를 쳐가며 더 만들어나갈 시간에 더 중요한 점수를 먼저 구현했어야 했었다. 그리고 스테이지는 2개도 정말로 많이 벅찼다. 혹시 나중에 후배가 유니티로 스테이지를 3개 이상 만들고 싶다고 하면 말리고 싶다.

4. 참고문헌

1. 절대강좌! 유니티5 : Unity 전문 개발자가 알려주는 효과적인 게임 제작 기법 (2015.07/이재헌 저)
2. Unity3dStudy.com (이재헌 저자의 사이트 Q&A)
3. 유니티짱으로 배우는 유니티5 3D 게임 제작 입문 (2015.11/마쓰다스 외 6명 저)
4. (개발 철학부터 원리와 스크립트로 완성하는) 열혈강의 유니티 게임 프로그래밍 (2014.07/주세영 저)
5. answers.unity3d.com/index.html (유니티 Q&A 공식 커뮤니티)
6. Stack Overfow (유니티 Q&A)
7. docs.unity3d.com/ScriptReference/MonoBehaviour.html (유니티 공식 Documentation으로 엔진에 있는 클래스 안의 여러 함수들과 그의 기능들, 쓰는 방식, 변수들 등 스크립팅하는 방법을 알 수 있다)
8. Youtube의 Unity 공식 계정이 올린 튜토리얼 영상들