소프트웨어융합개론 최종 보고서

**Crome Arena :**

**Photon Engine**을 이용한

성장형 멀티플레이 격투게임

소프트웨어융합학과 이승규 2019102115

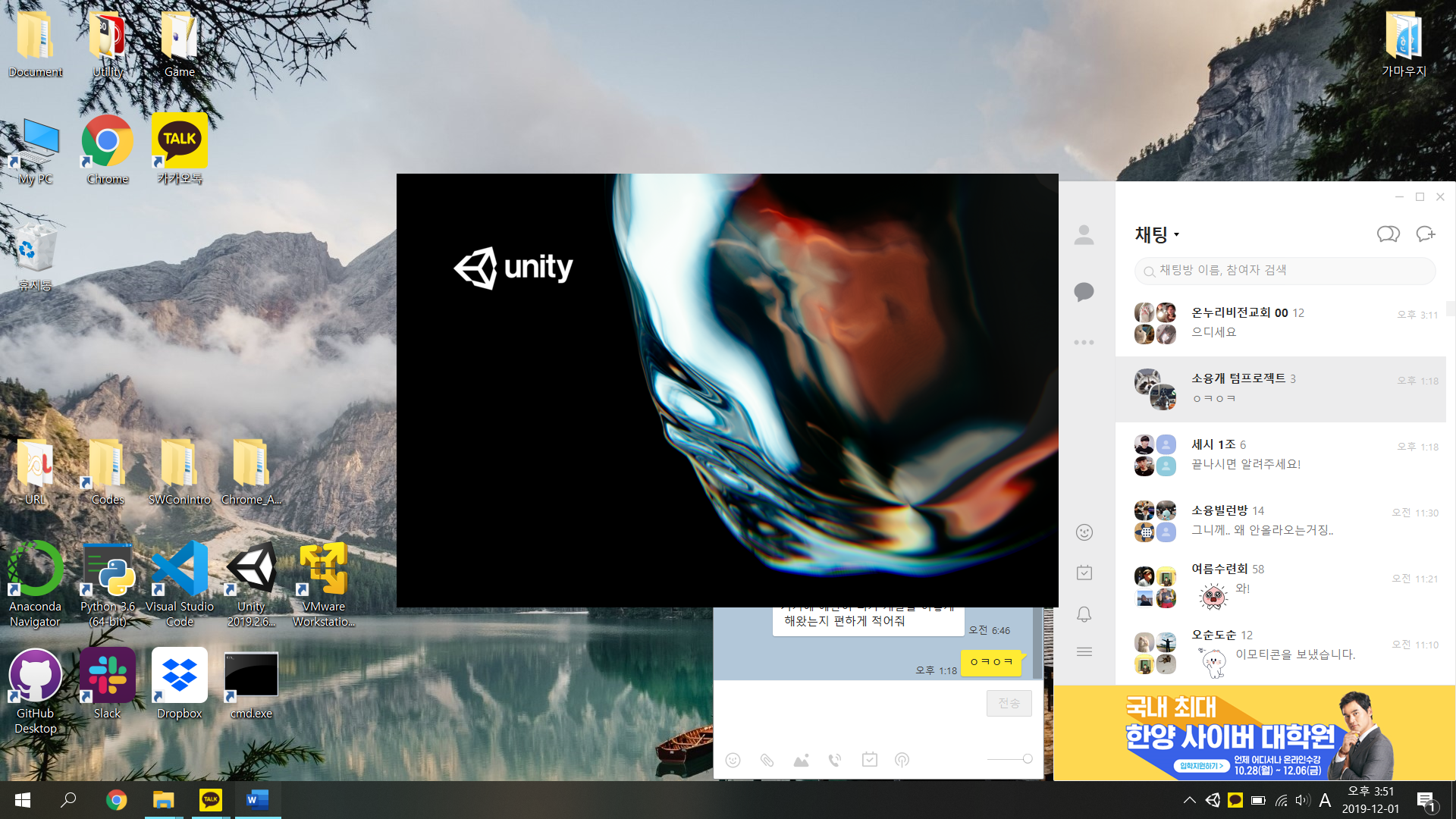
소프트웨어융합학과 이예찬 2019102117

소프트웨어융합학과 이륜하 2019102114

1. 주제 선정 배경

오늘날 PVP장르의 게임들을 살펴보면 순발력과 같은 개개인의 신체적 능력에 따라 게임 플레이에 막대한 영향을 끼치는 것을 볼 수 있다. 특히 철권과 같은 격투장르와 배틀그라운드 같은 배틀로얄 장르는 플레이어의 신체적 능력이 게임의 승리에 90%, 게임의 이해도가 10%정도 미치는 등 신체적 능력이 게임의 승패를 많이 가른다. 하지만 사람의 신체적 능력은 개개인마다 다른 것이 당연하기 때문에 출시한지 어느정도 시간이 지난 게임들은 신체적 능력에 따른 진입 장벽이 생겨 신규 유저가 안정적으로 그 게임을 플레이하기 어려울 때가 많다. 그래서 신체적 요소를 제외하고 성장적 요소를 넣어 다른 플레이를 보여줘 신체적 차이를 줄일 수 있는 게임을 주제로 선정하게 되었다.

1. 배경 지식
2. Unity game engine



본 프로젝트는 유니티 게임 엔진과 유니티에서 제공하는 유니티 에셋 스토어에서 판매하는 소스들을 기반으로 한다.

유니티는 3D 및 2D 비디오 게임의 개발 환경을 제공하는 게임 엔진이자, 3D 애니메이션과 건축 시각화, 가상현실 등 인터랙티브 콘텐츠 제작을 위한 통합 제작 도구이다.

1. Photon Engine



Photon Engine은 Photon Unity networking으로 유니티 게임엔진에서 멀티플레이를 할 수 있게 해주는 패키지이다. 유니티 에셋 스토어에서 다운로드가 가능하며 현재 Photon PUN 2 무료 에디션과 2+인 유료버전이 있다.

Photon Server 멀티플레이어 API를 이용하여 여러 플레이어들이 공유된 게임 세션에 참여하고 플레이어들의 플랫폼에 상관없이 접속한 플레이어 간에 데이터와 메시지 전송을 실시간으로 하며 동기화를 가능하게 해준다.

Photon 클라우드는 클라이언트 SDK를 이용하여 다양한 장르의 게임을 개발할 수 있다. 모든 어플리케이션은 C# 소스코드가 제공되며 성능에 최적화되어 있어 필요에 따라 커스터마이징이 가능하다.

1. Promotion, WavePad(데모버전), Clip Studio

Game Scene의 배경, 캐릭터와 캐릭터 애니메이션, Effect sound, background music등을 디자인할 때 사용된 기타 프로그램들이다.

Promotion은 도트를 찍을 때 주로 쓰는 툴 프로그램으로 캐릭터들의 디자인과 스킬 디자인, 그리고 아이템 디자인에 사용되었다.

WavePad(데모버전) 스킬들의 효과음을 편집하기 위해 사용한 툴 프로그램이다.

Clip Studio는 게임 씬의 배경(로고, 인 게임의 지형 등)을 디자인하기 위해 사용한 프로그램이다.

4) Unity Collab

Unity에서 제공하는 Unity 전용 협업 툴이다. 이를 통해 자신이 개선한 코드, 씬 등 변경 내용들을 손쉽게 업로드/다운로드할 수 있으며, 버그 등 문제가 있을 시에는 이전 버전으로 백업도 가능하다. 이 툴을 쓰며 커뮤니케이션함으로써, 파일을 주고받는 것에 비해 개발 진행단계를 실시간으로 간편하게 확인하고 각자가 해야 할 일을 명확하게 알 수 있었다.

1. 개임 개요

장르 : 성장형 2D 배틀 로얄

진행 : 게임 시작 후 플레이어들 끼리 자신의 캐릭터의 스킬을 사용하면서 경쟁. 맵 곳곳에 무작위로 떨어지는 아이템(배터리)을 획득하여 캐릭터의 hp를 회복하고 능력치를 소폭 상승시킴.

승리 조건 : 스킬들을 이용해 맵 내의 자신 이외의 모든 상대방의 hp를 0이하로 감소 시 승리

캐릭터 :

1. Sword

검을 주무기로 사용하는 근접형 캐릭터

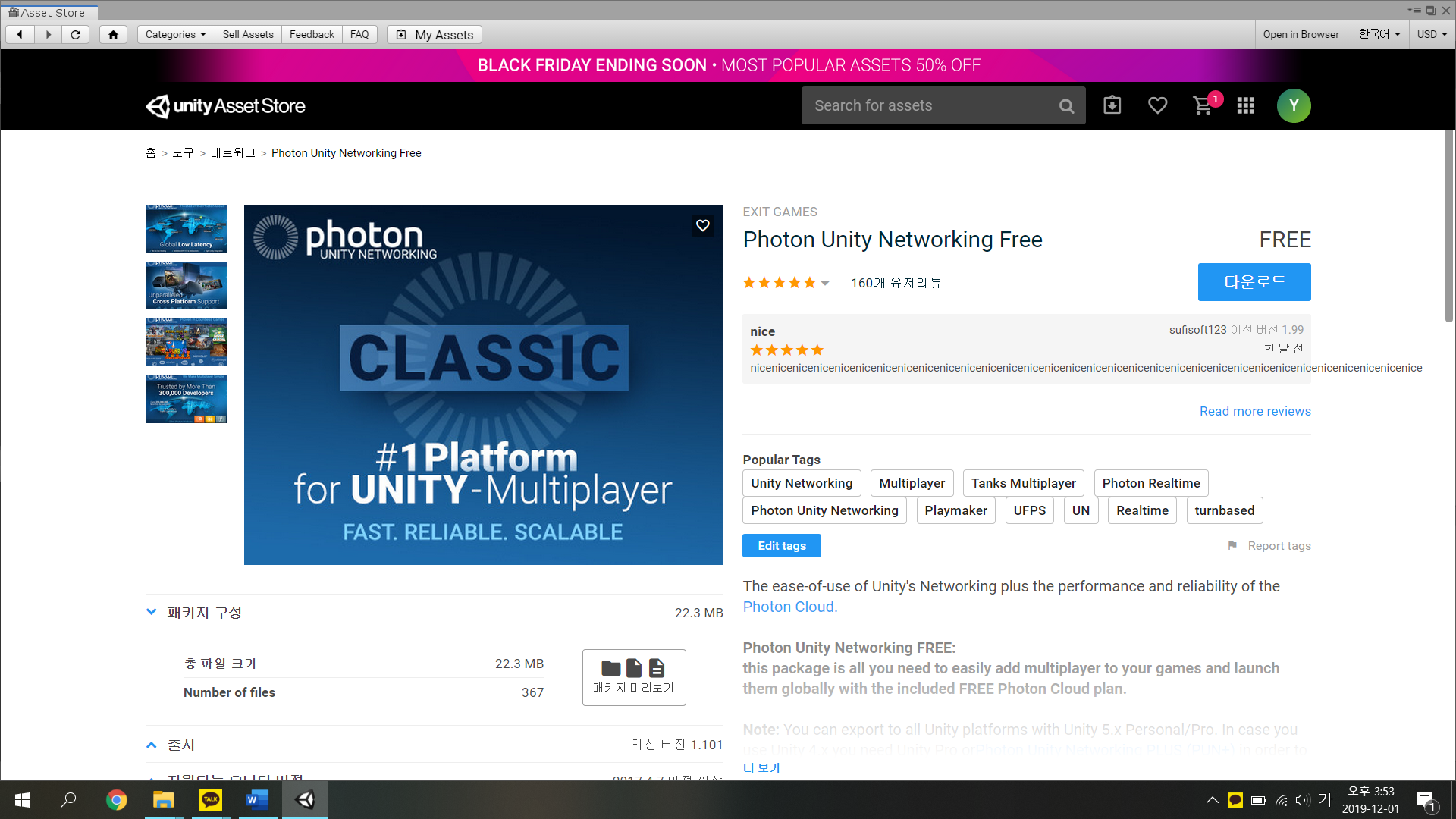
1. Shield

쉴드의 에너지를 사용하여 원거리 공격을 사용하는 캐릭터

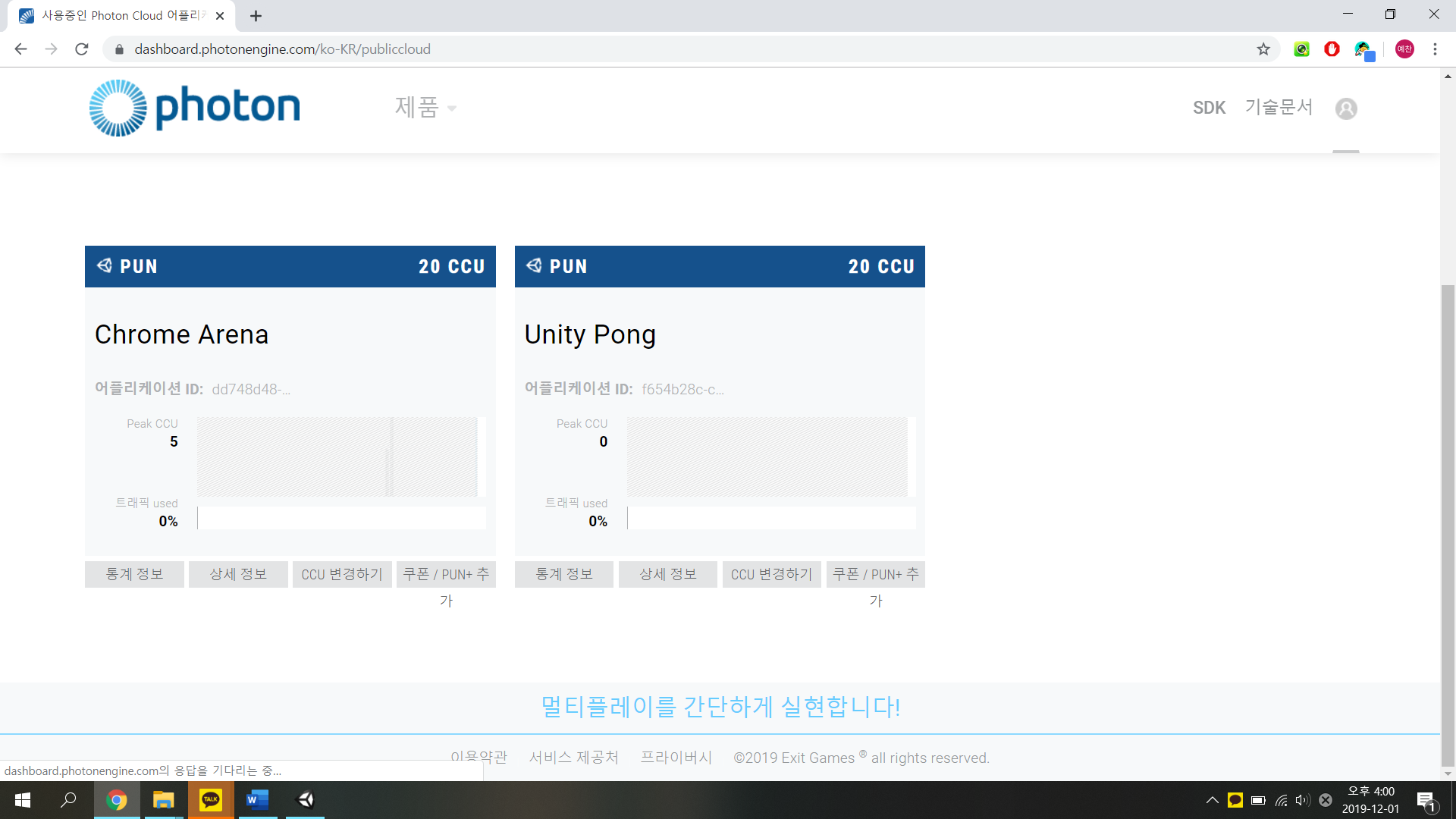
아이템(배터리) : 게임 내 hp 회복, 공격력 상승, 방어막 회복 등 여러 순기능을 캐릭터에 적용시켜주는 오브젝트

1. Hp회복
2. 공격력 상승
3. 방어막 회복
4. 최대 Hp 상승
5. 점프력 상승
6. 개발 과정
7. 멀티플레이 환경 구축

- 유니티 에셋 스토어에서 PUN 패키지를 다운로드한다.



- [www.photonengine.com](http://www.photonengine.com)에 접속하여 회원가입한 뒤 photon cloud application을 생성한다.



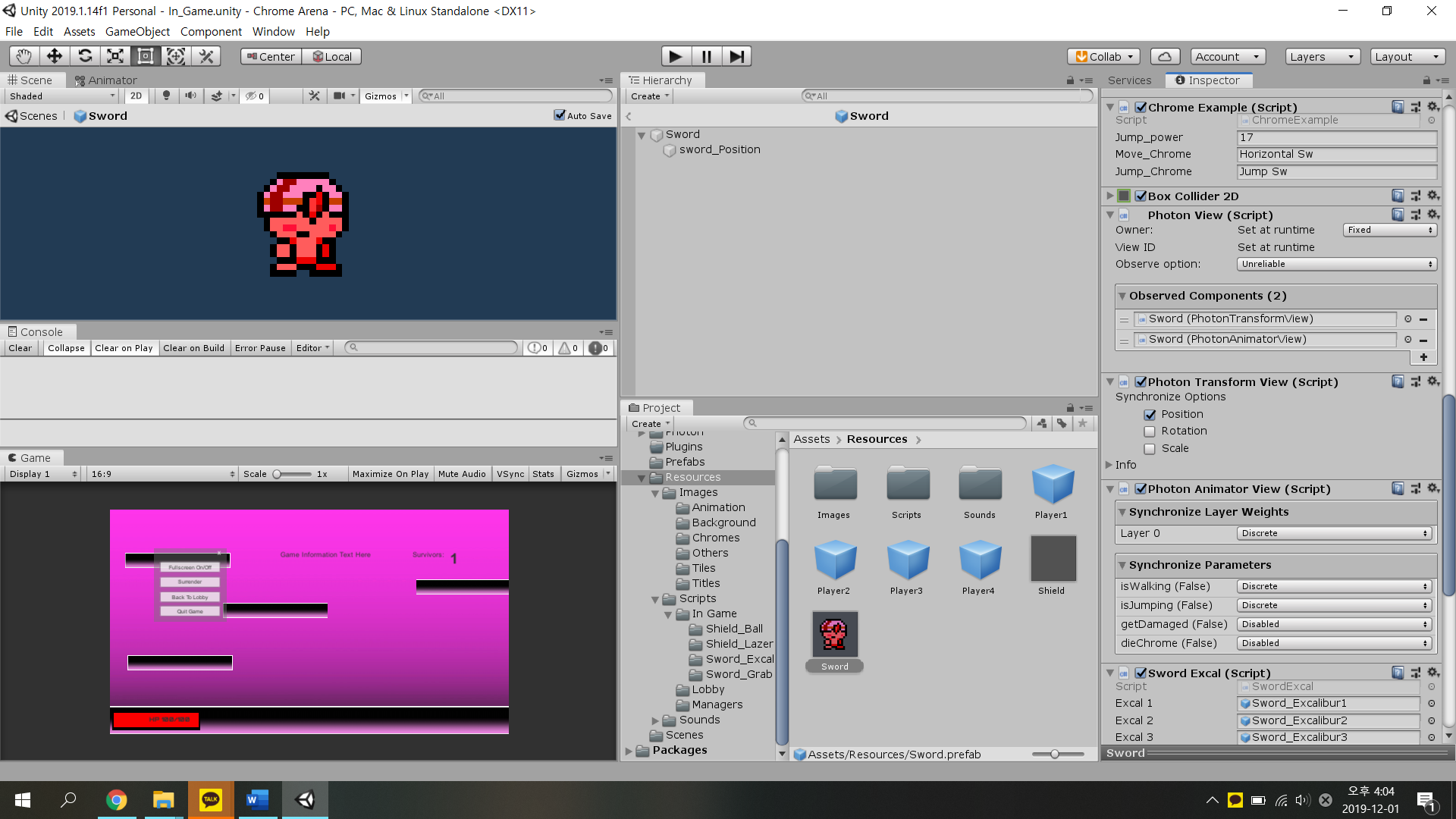
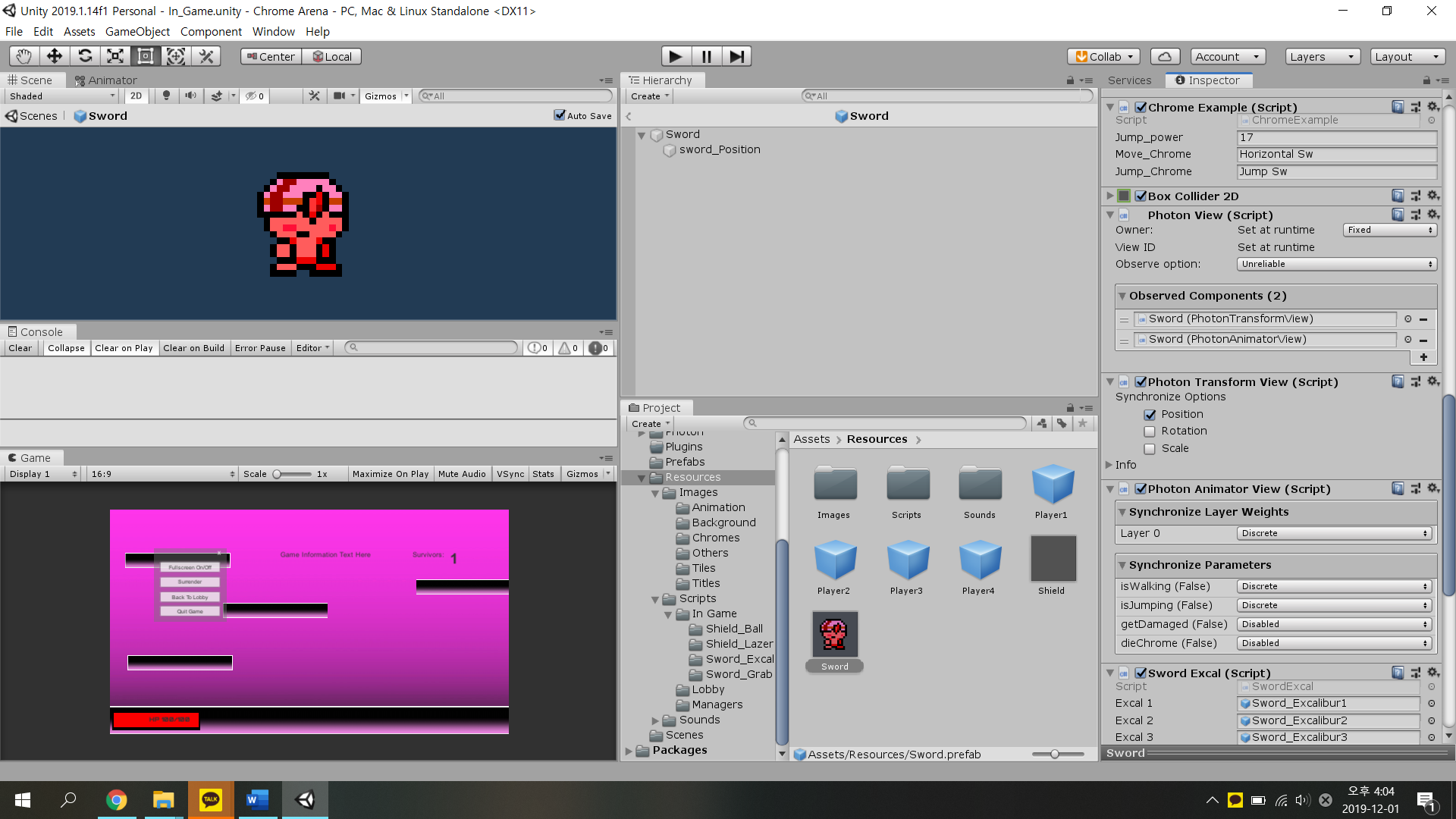
- application에 할당된 application ID를 개발할 Unity project에 등록하여 그 project가photon cloud에 의해 인식되어 cloud의 리소스를 사용할 수 있도록 설정한다.

- 다음 기능을 코드로 짠다:

호스트와 클라이언트가 마스터 서버에 접속하게 함

호스트가 게임 룸을 만들어 클라이언트가 접속하게 함

마스터 서버 접속 실패, 방 제작 실패 등 예외사항 처리



- Prefab화한 게임 캐릭터 오브젝트에 각각 Photon View, Photon Transform View, Photon Animator View 컴퍼넌트를 추가하여 오브젝트의 트랜스폼 정보(위치, 회전, 크기)와 오브젝트의 애니메이션 정보가 네트워크상으로 전달되도록 한다.

- Prefab의 애니메이션을 담당하는 C# script를 photon이 인지하게 만듦으로써 photon network에 전달할 animator parameter를 설정한다.

1. 멀티플레이의 원리

이 부분에 대해서는 간략하게 소개하겠다.

본 프로젝트에서는 Photon.PUN에서 지원하는 PhotonNetwork를 이용해 플레이어들을 마스터 서버에 접속시키고, 호스트가 게임 룸을 생성하면 나머지 클라이언트들이 호스트가 만든 룸에 접속함으로써 같이 게임을 시작하는 방식으로 개발되었다.

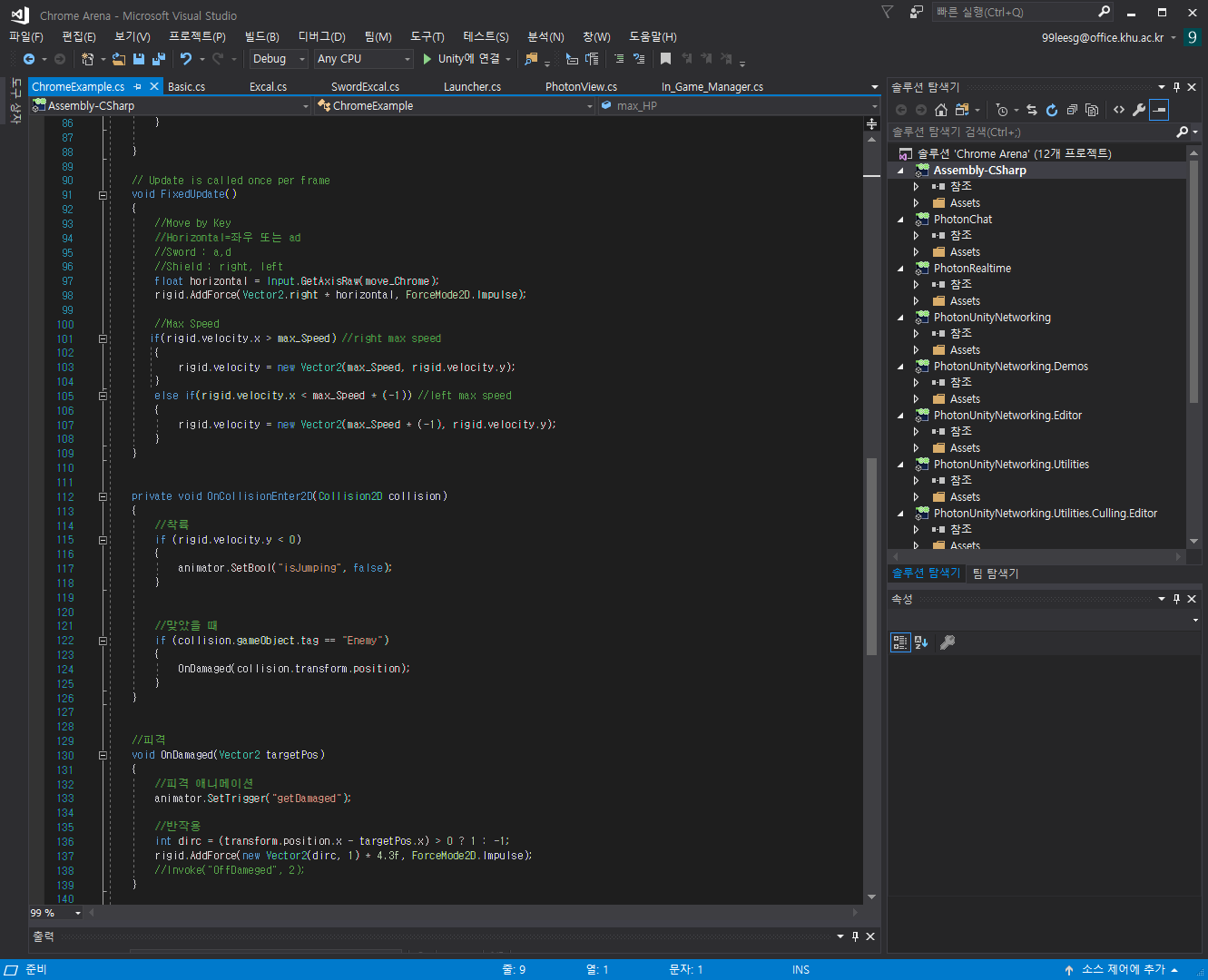
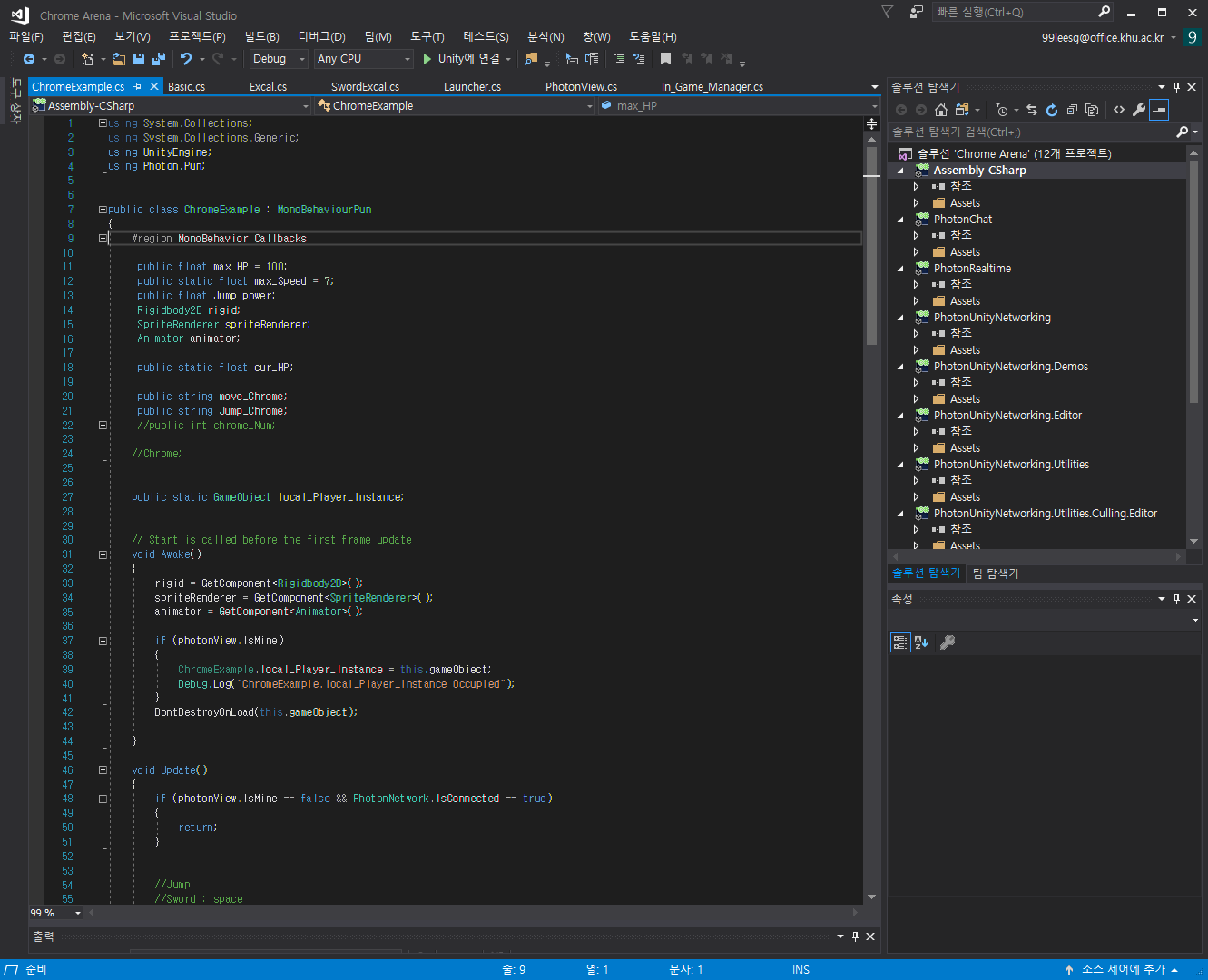
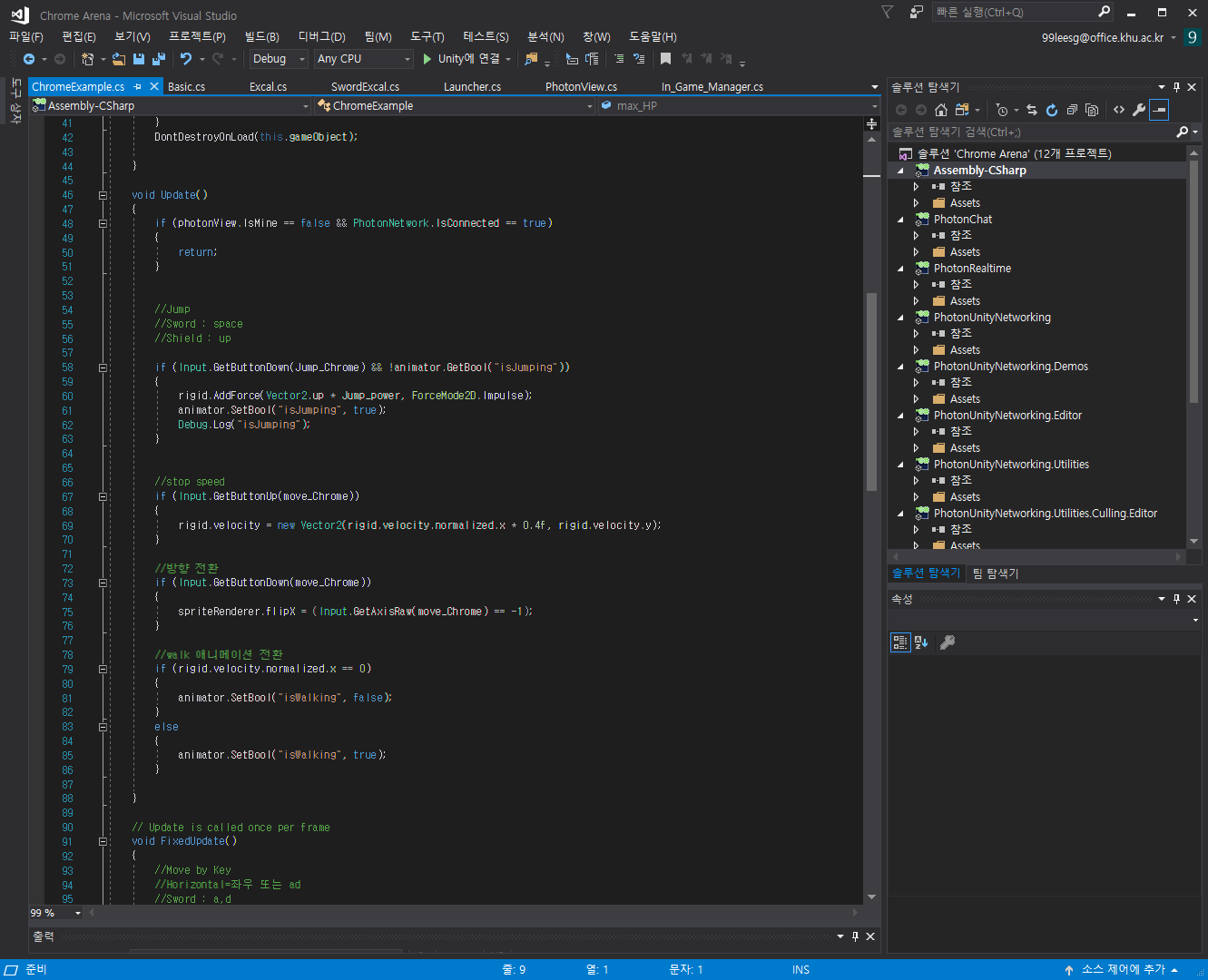
이 때 게임상의 오브젝트들은 컴포넌트 PhotonView.cs가 Photon animator view(애니메이션을 관측), photon transform view (이동, 회전, 크기 3가지요소 트랜스폼 관측)에서 관측된 데이터를 호스트에게 넘긴다. 그리고 호스트는 그 데이터를 받은 후 호스트의 룸에서 연산을 마친 뒤 그 결과를 다른 클라이언트들에게 알려준다.

이를 통해 다른 플레이어의 이동, 배터리(아이템)를 획득 여부, 애니매이션 상태정보 등 여러 정보를 클라이언트가 받게 된다.

1. 게임 시스템

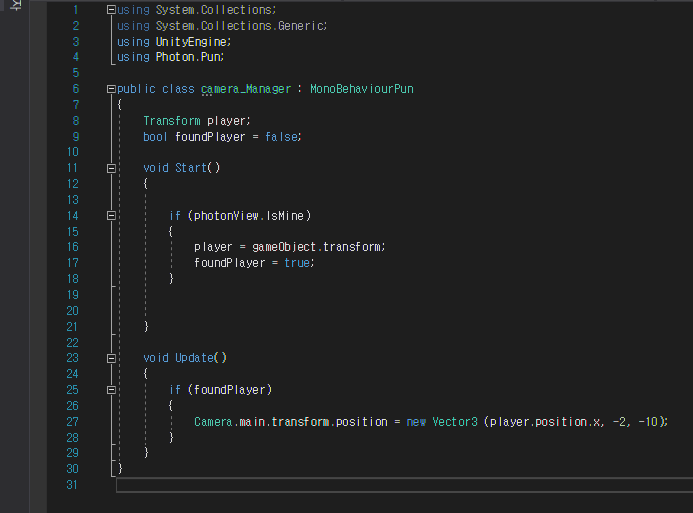
개발하기 전 멀티플레이를 구현하고 코드를 짜는 것이 좋다는 조언을 듣고 멀티플레이에 영향을 받지 않는 기본적인 게임 시스템이나 캐릭터의 움직임부터 코드를 작성하고 계획을 하였다. 그리고 빠진 요소를 찾을 때마다 새로운 스크립트를 만들어 스크립트들 간의 구조가 복잡해지고 변수를 통일하는 데에도 문제점이 생겼다. 그로 인해 시스템 관련 스크립트만 6개가 되었다.(PUN관련 스크립트를 제외하고)

1. 캐릭터 컨트롤 스크립트(ChromeExample.cs)



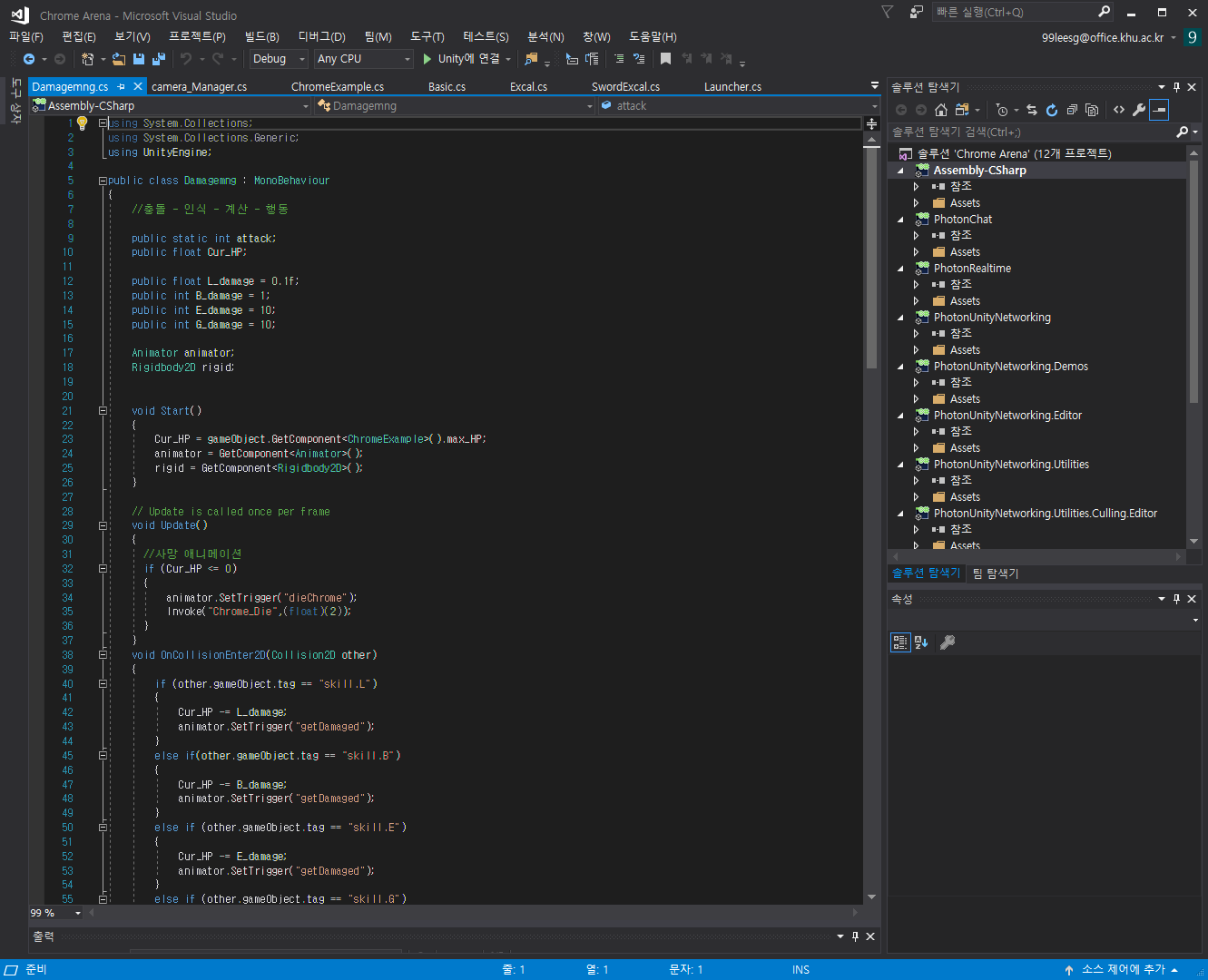
캐릭터의 기본적인 움직임을 위한 스크립트다. 멀티플레이의 구현 전 캐릭터간의 상호작용(전투)을 테스트하기 위해 움직임을 한 키보드에 할 수 있도록 코드를 작성했다.

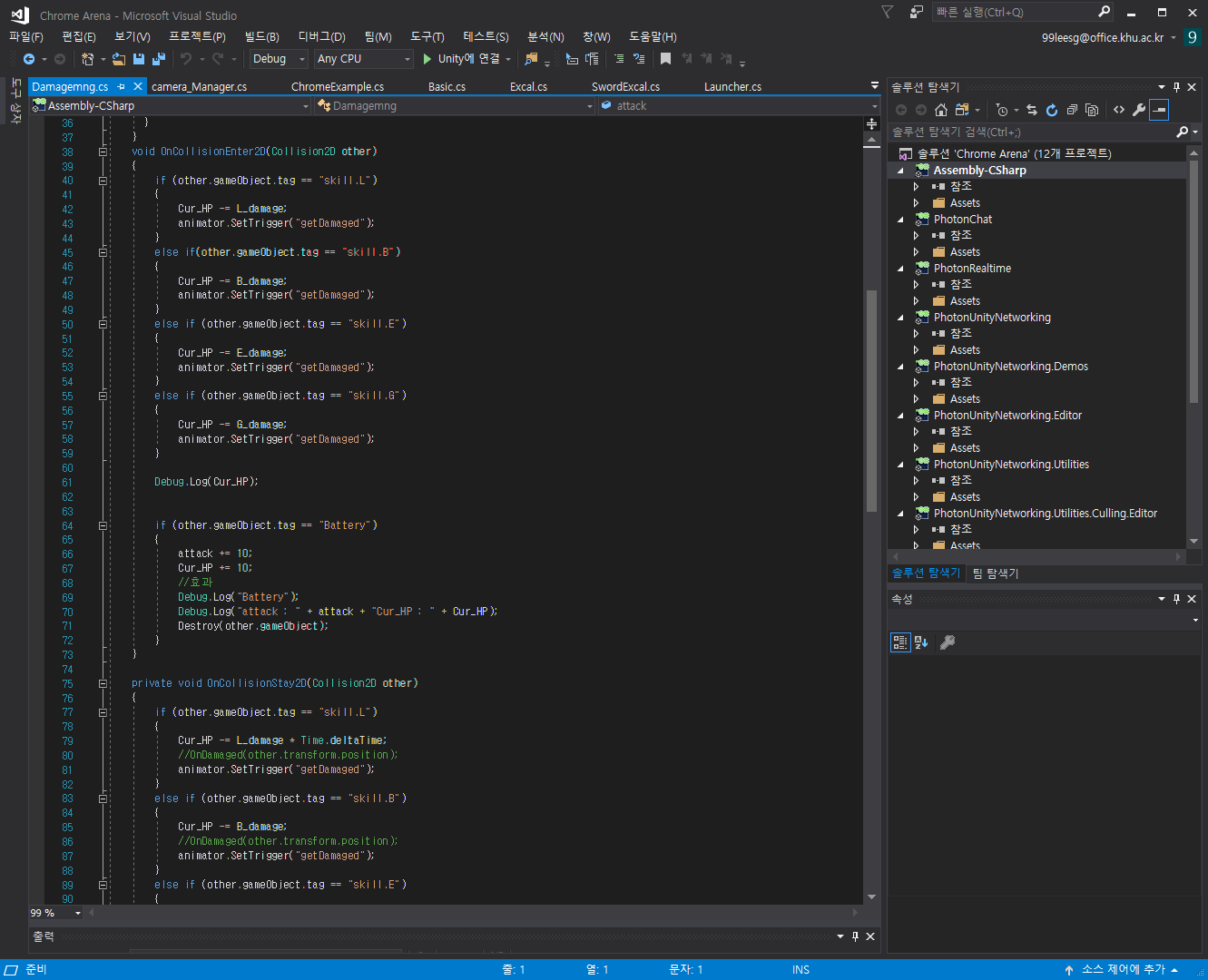
1. 시점이동 스크립트(camera\_Manager.cs)

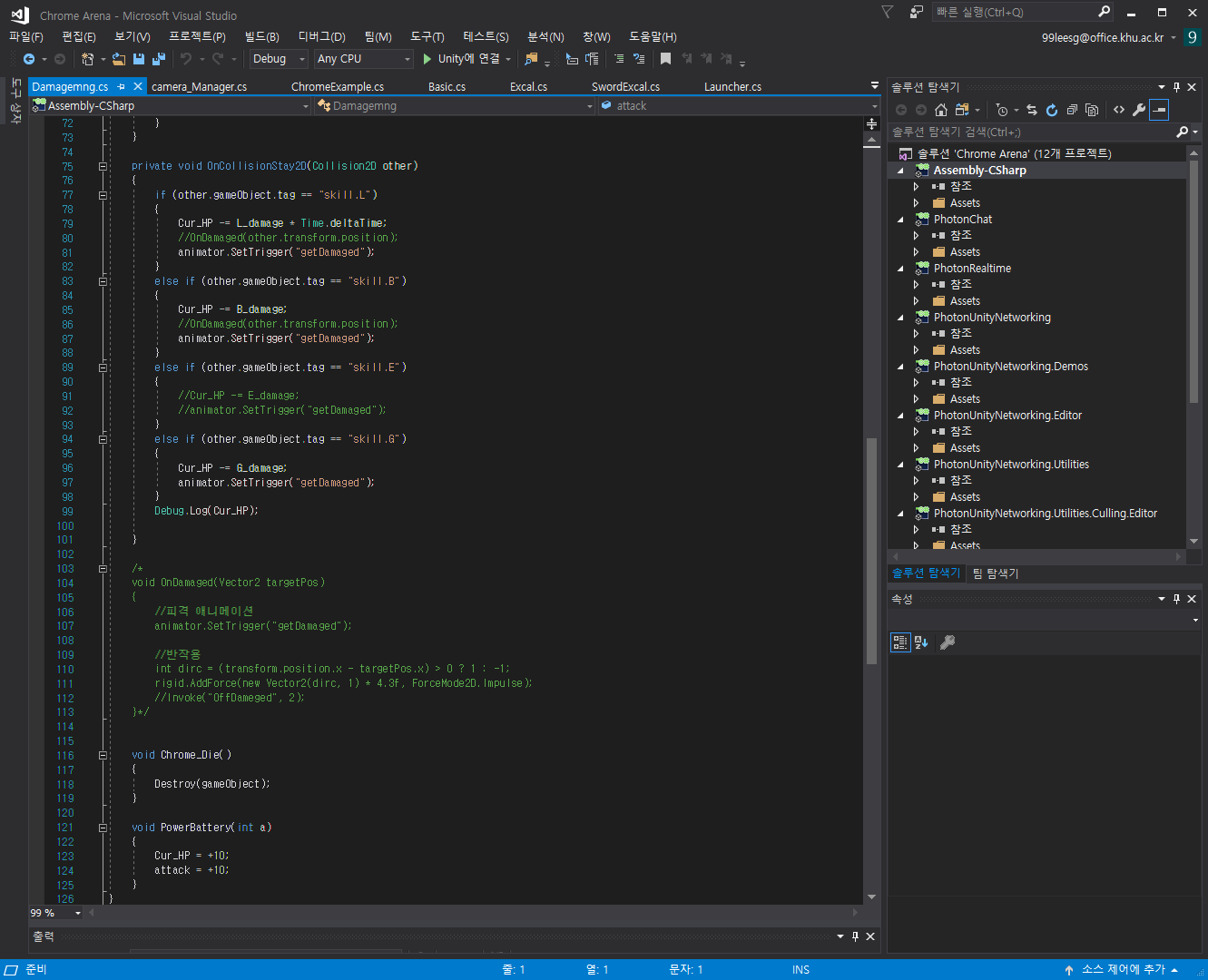


카메라가 캐릭터를 따라 움직이게 하는 스크립트.

1. 데미지 계산 스크립트(Damagemng.cs)

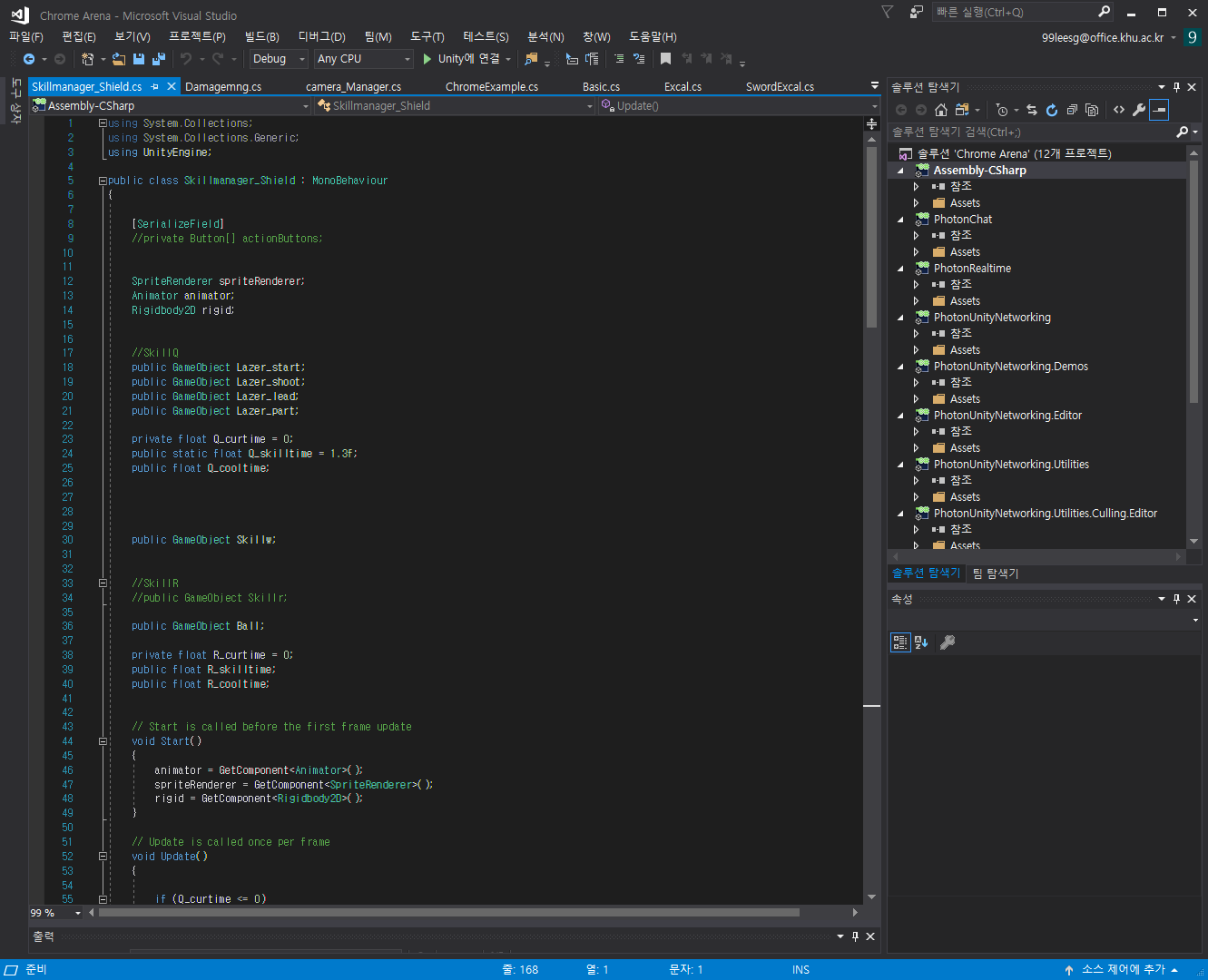


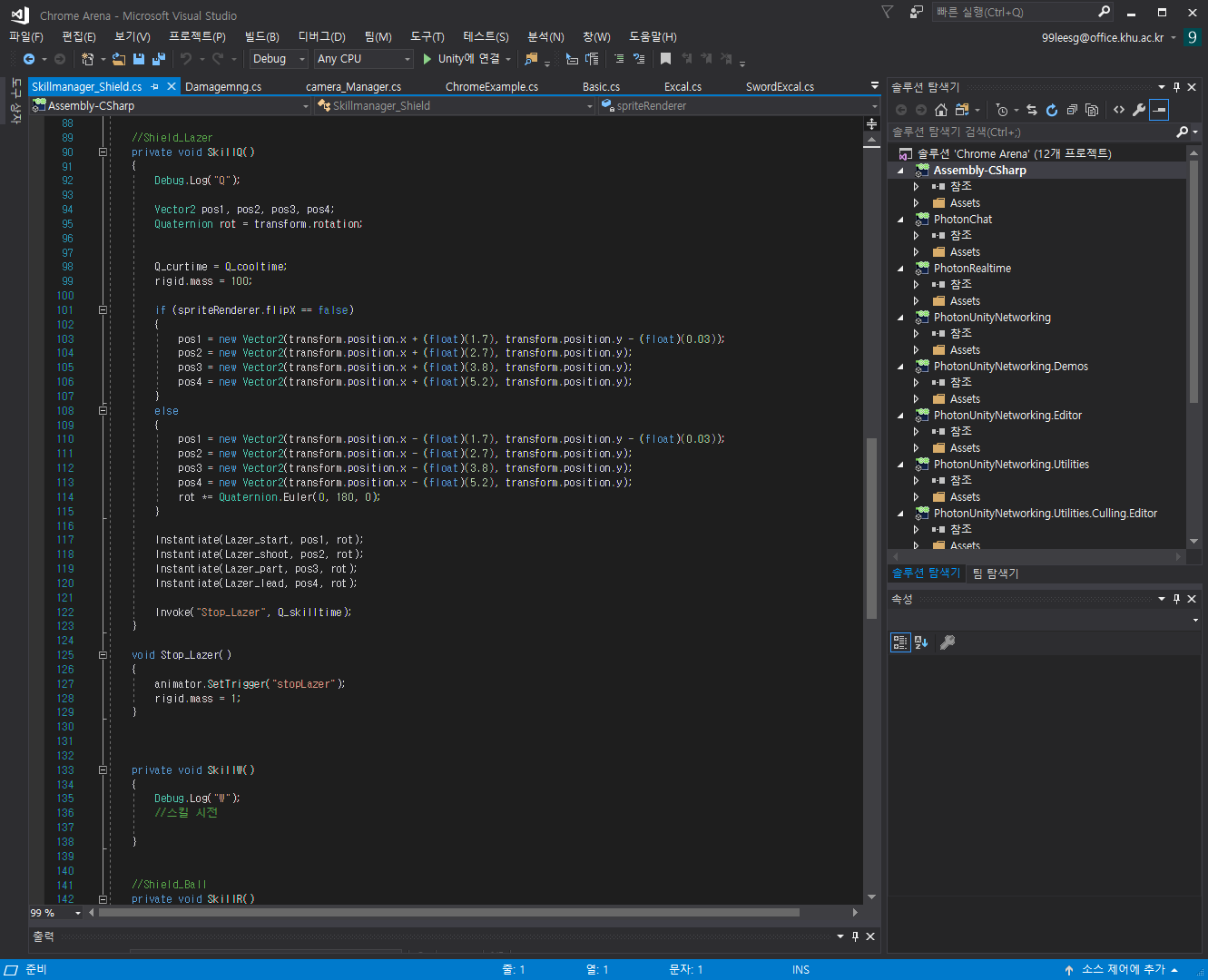


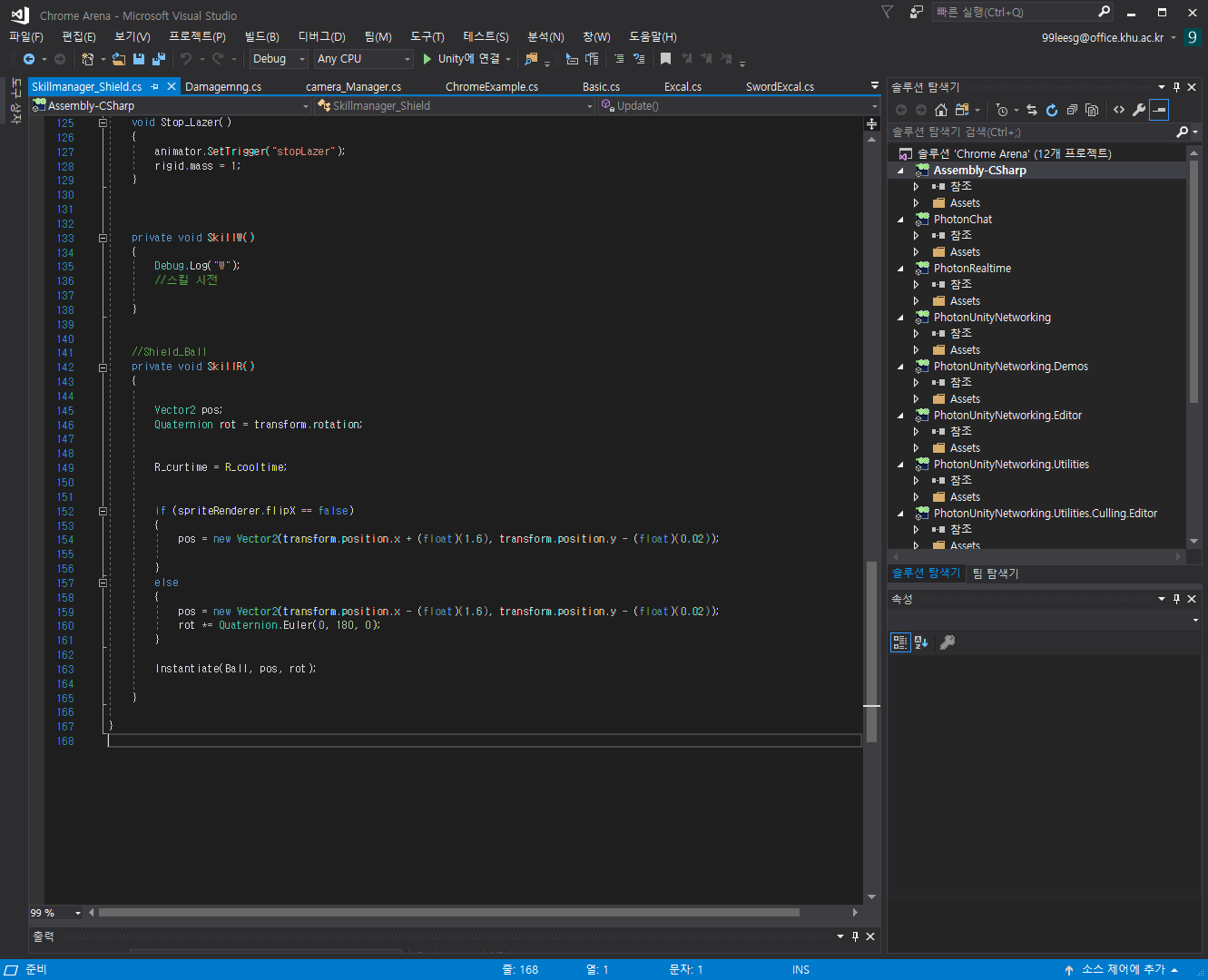


In\_game씬에서 캐릭터간의 전투에서 데미지를 주고받게 하고 사망을 구현한 시스템 스크립트.

1. 캐릭터 스킬 매니티먼트 스크립트(Skillmanager\_Shield.cs, Skillmanager\_Sword.cs)

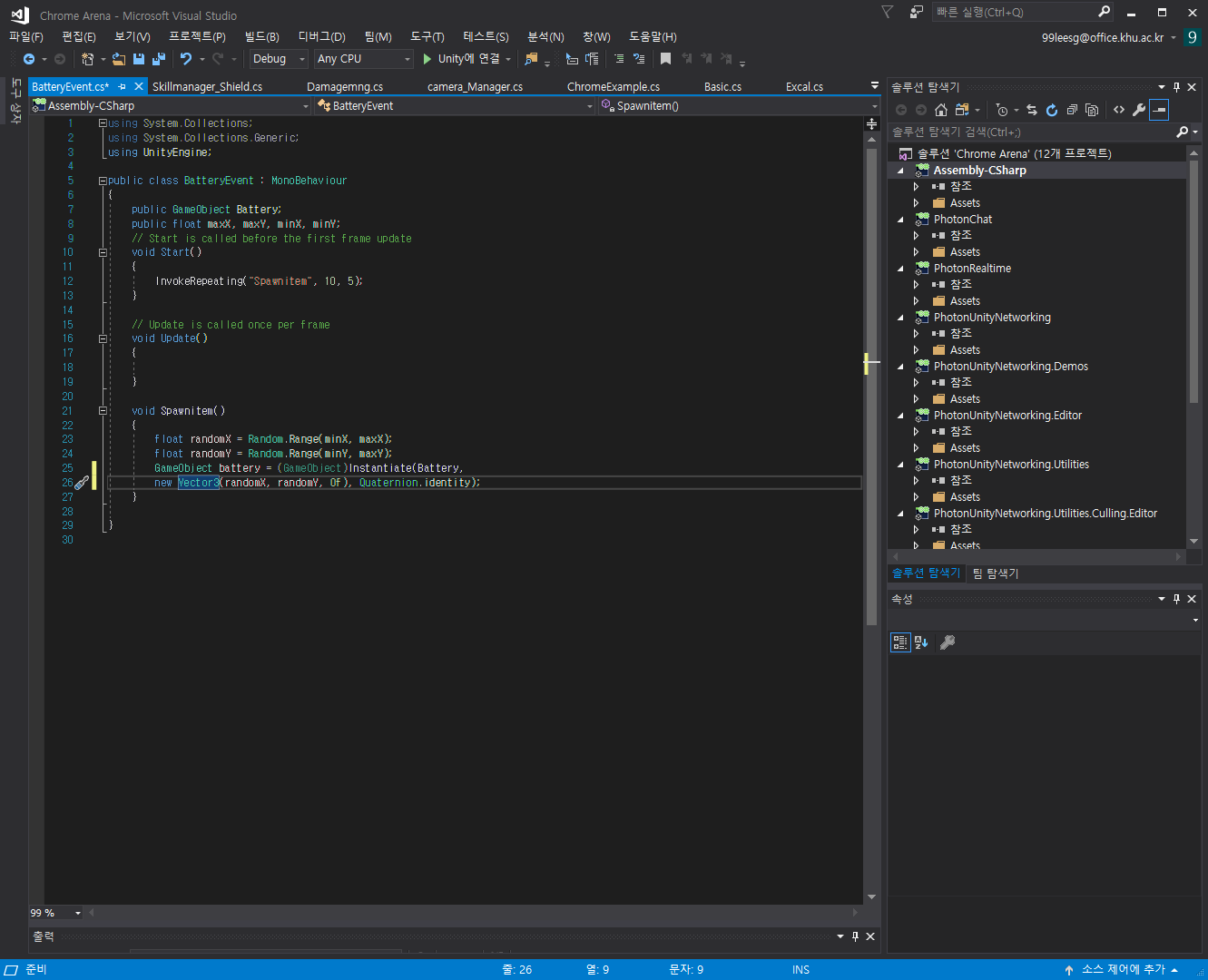






(Skillmanager\_Sword.cs도 위와 같은 형태로 작성되어 있다.) 캐릭터 하나의 스킬들을 관리하는 스크립트다. 원래 스킬을 프리팹화 하여 캐릭터 공통으로 전역 변수로 스킬 단축키 지정을 편하게 하려 했으나 구현 과정에서 문제가 생겨 캐릭터 별로 다른 매니지먼트 스크립트를 작성했다.

1. 아이템 스폰 이벤트 스크립트(BatteryEvent.cs)

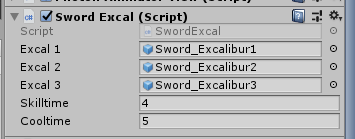


1. 캐릭터의 애니메이션 및 스킬 구현

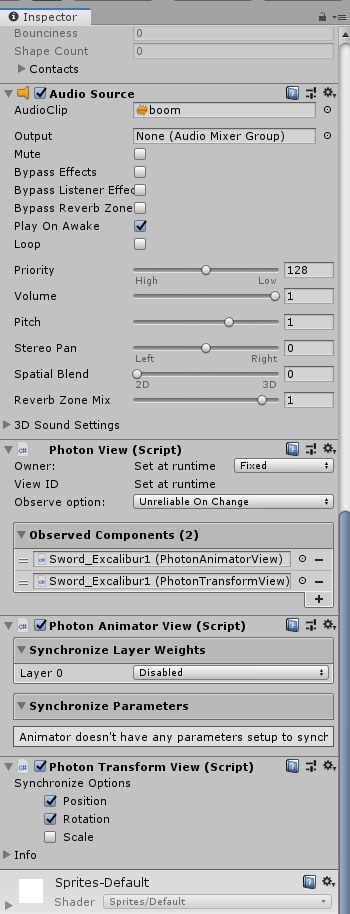
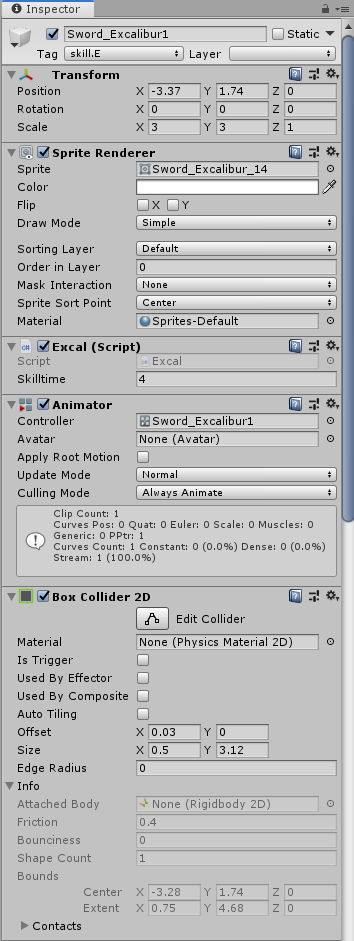
캐릭터들의 스킬들은 스크립트 형태로 작성하여, 스크립트 내의 전역변수에 프리팹화한 스킬의 애니메이션과 함수들을 넣어 완성했다. 그리고 완성한 스킬들과 GameManager와 같은 시스템에 대한 스크립트들을 넣어 캐릭터와 아이템들을 프리팹화했다.

예시) 캐릭터 Sword의 스킬 Excalibur

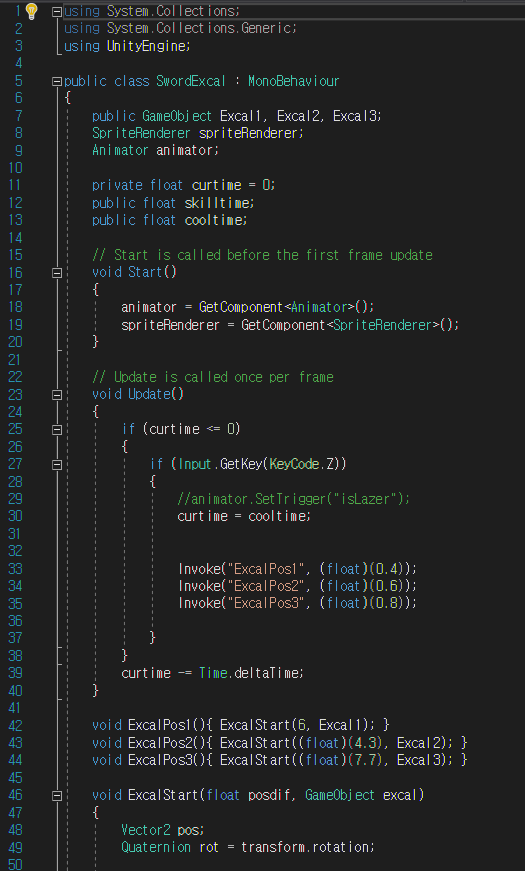
스킬 컴포넌트 SwordExcal.cs



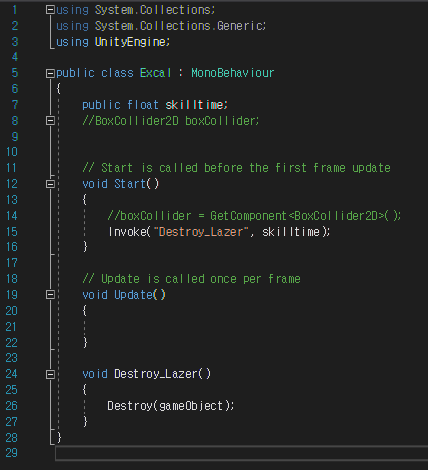
프리팹 Sword\_Excalibur1의 컴포넌트(Sword\_Excalibur1 ~ 3은 모두 동일하다.)



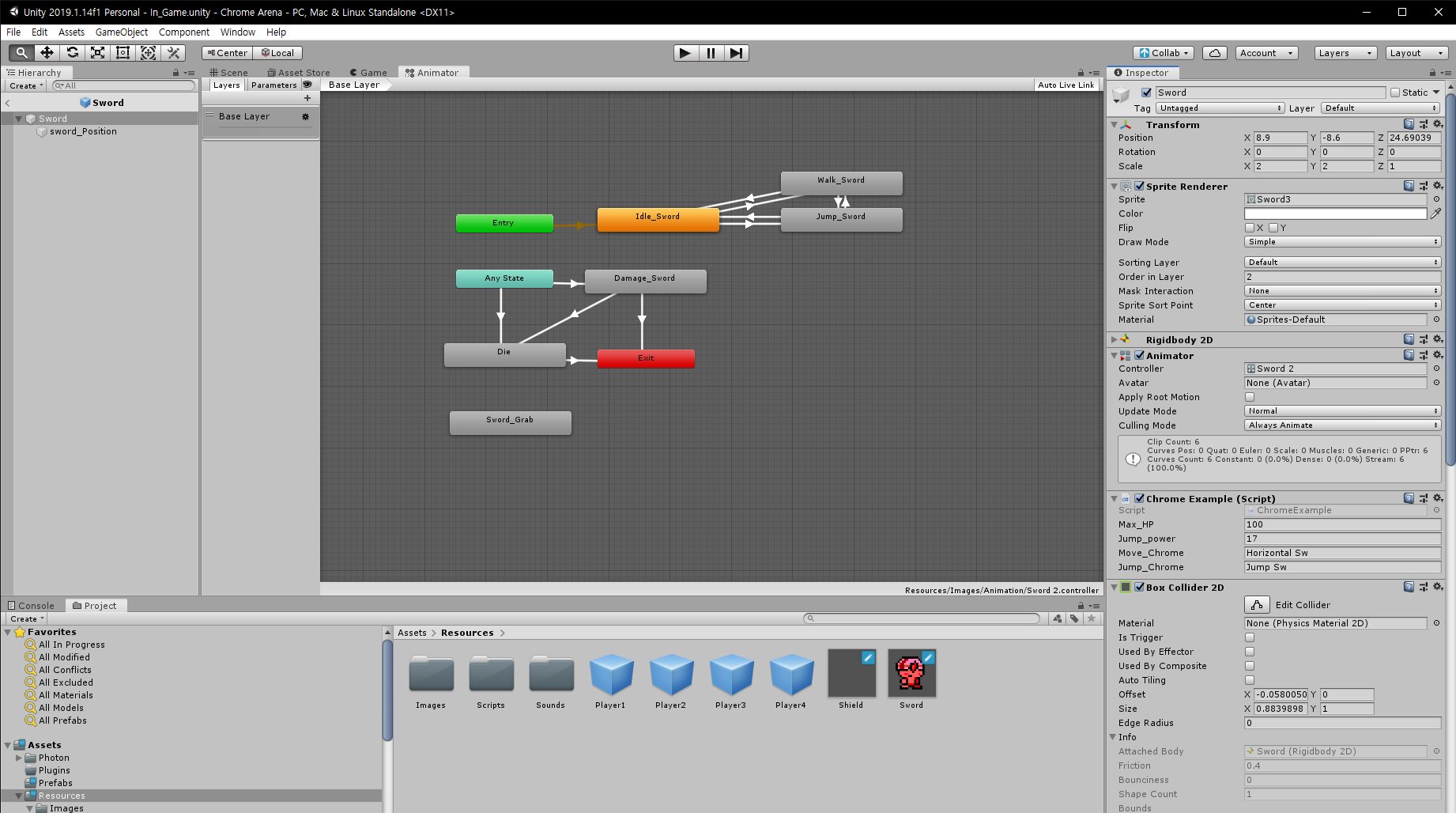
SwordExcal.cs 스크립트 코드



Sword.cs 스크립트 코드



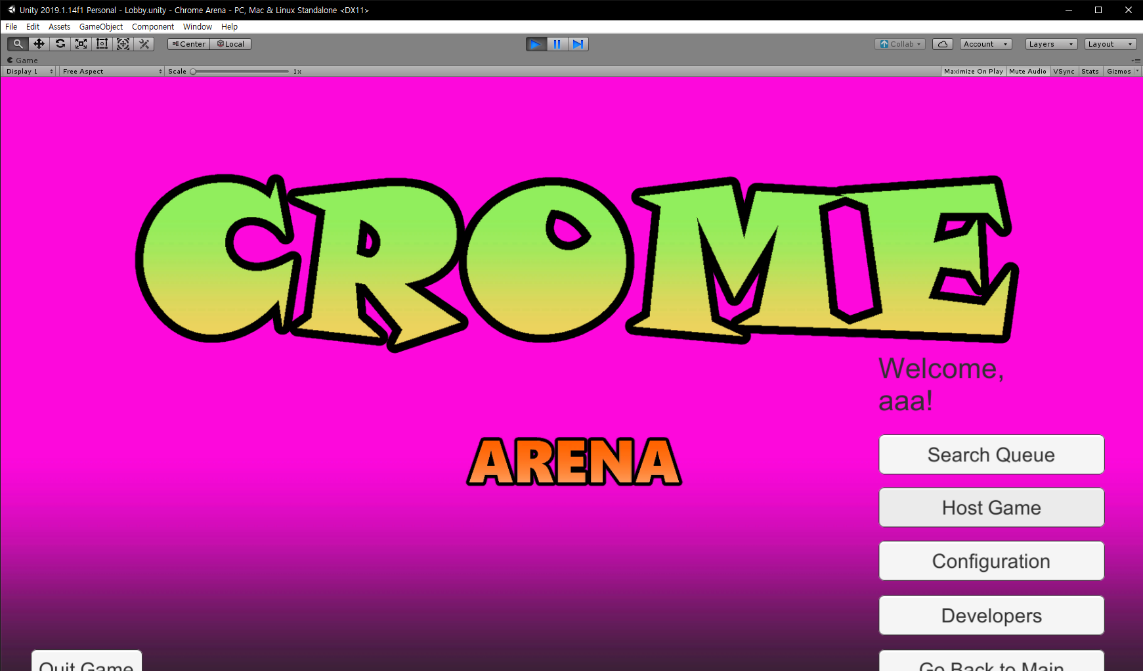
예시2) 캐릭터 Sword 프리팹(애니매이터)



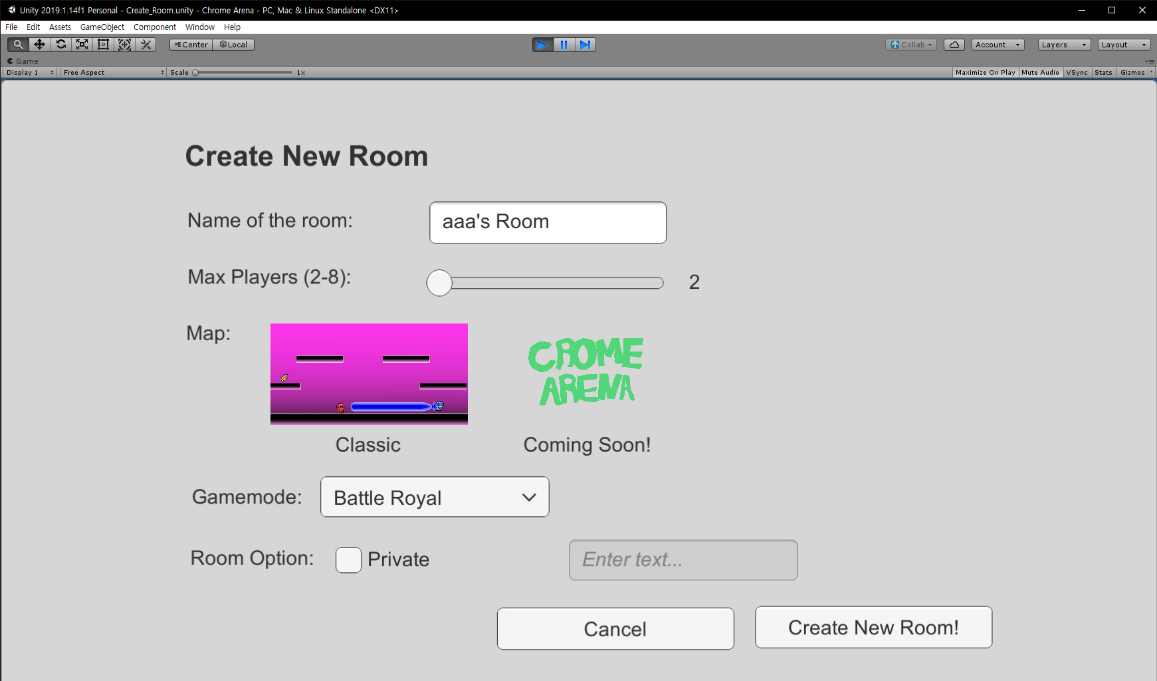
1. 실행



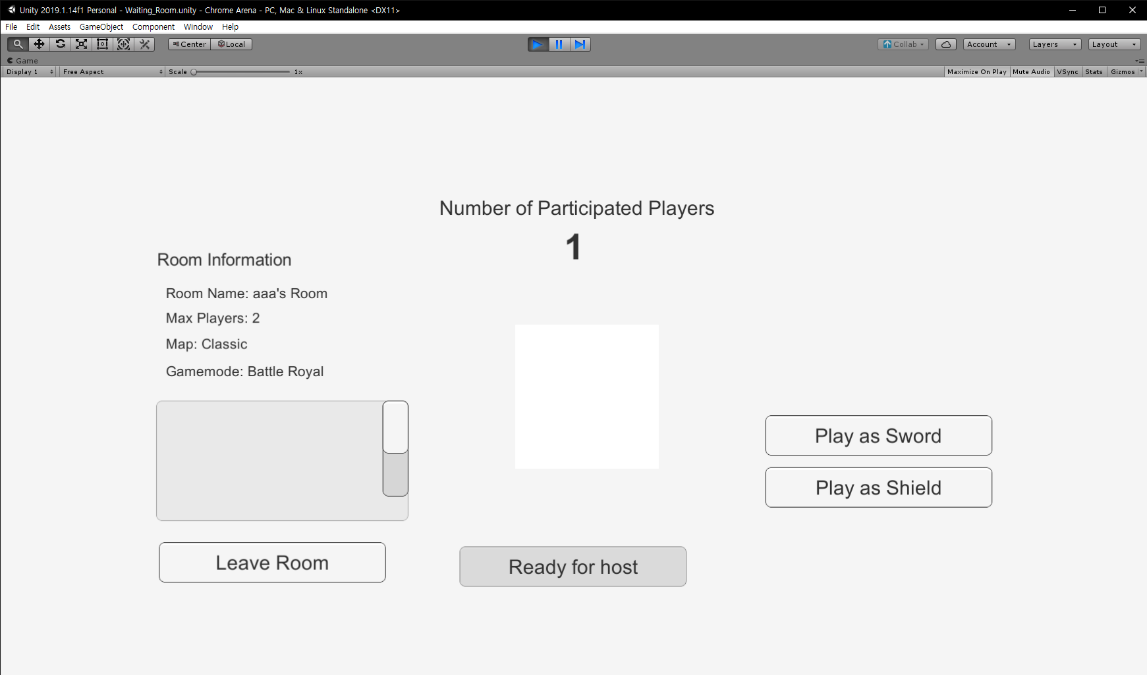
1단계 닉네임(예시, aaa)을 입력하고 PUN 마스터 서버 접속

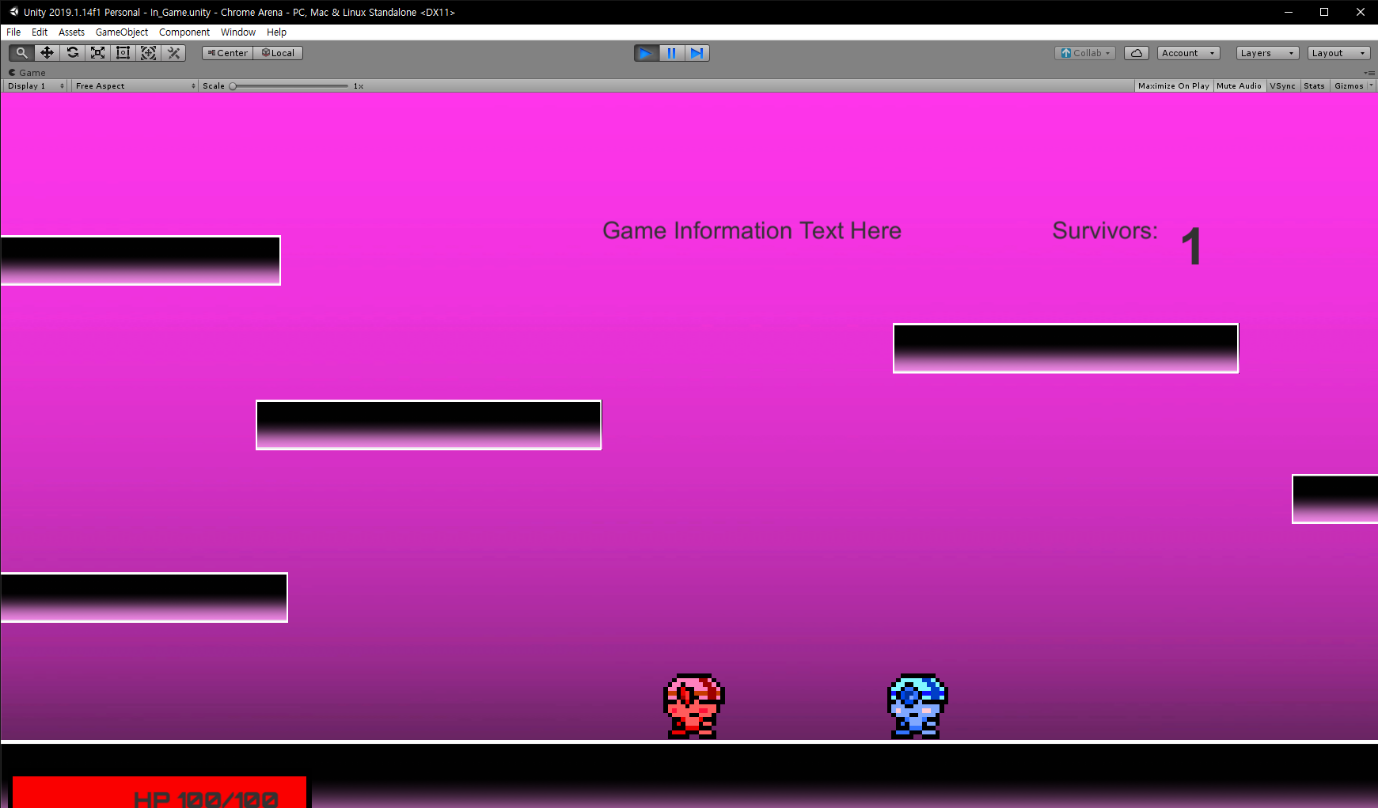


2단계 서버 접속 후 Host Game버튼을 통해 방을 직접 생성하거나, Search Queue를 통해 마스터 서버에 생성된 방에 들어간다.



3단계 Host Game 버튼을 눌렀을 때 생성할 방을 설정 최소2인에서 8인까지 가능하며, 방 이름을 새로 지을 수 있다.



4단계 Host Game을 통해 방을 생성하거나 Search Queue를 통해 생성된 방에 입장할 경우 나타나는 방 대기실. 가운데 숫자가 대기실 인원수다. 게임 시작 전, 오른쪽의 캐릭터 선택을 통해 플레이할 캐릭터를 선택하고 호스트가 아닌 사람은 Ready for host 버튼은 누르면 룸 호스트(Host Game을 클릭해 방을 생성한 플레이어)에게 Ready for Game버튼이 생기며, 게임을 시작한다.  


5단계 게임 플레이 화면이다. 스킬 테스트를 위해 원래 캐릭터가 아닌 다른 캐릭터를 생성해 놓았다.