3. Design

22112594 차아현
ahinc0218@gmail.com

[Revision history]

Revision date	Version #	Description	Author
	0.1	First document	

Yeungnam University

= Contents =

1. Introduction

· Summary

한국에서 10명 중 6명은 취미로 게임을 하고 있다. 생존 시뮬레이션 게임을 플레이하면서 다소 비현실적인 요소들이 있어 몰입감이 떨어지는 문제가 있었다. 이에 사실적이고 현실적인 시뮬레이션 게임을 좋아하는 사람들을 위해 게임 'Where rest is none'을 구상하게 되었다.

· Introduction of 'Where rest is none'

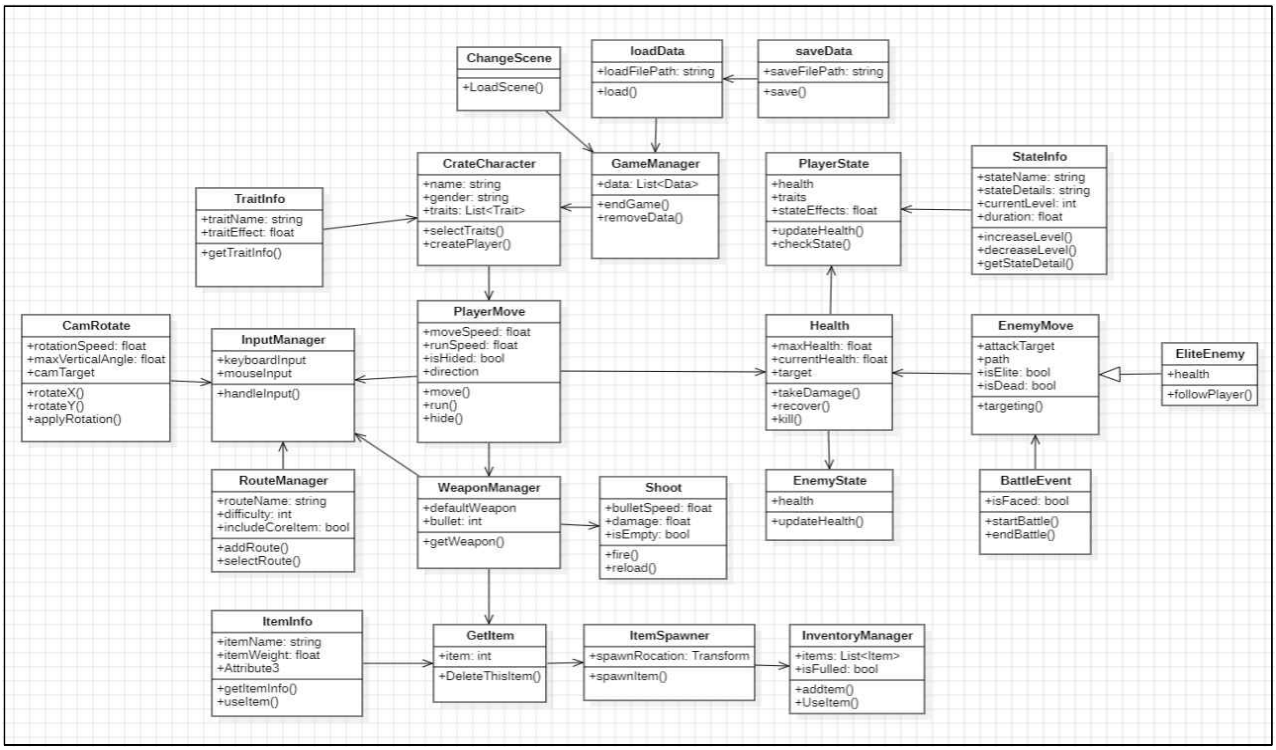
이번에 제작하게 된 게임 'Where rest is none'은 3인칭 시점의 생존 시뮬레이션 게임이다. 해당 게임은 스텔스 방식의 플레이를 기반으로, 플레이어는 최대한 적과의 직접적인 전투를 피해야 한다. 적에게 발각될 경우 고난이도의 전투 이벤트가 발생하며, 이는 생존 실패로 이어질 수 있기 때문에 이 점을 고려하며 게임을 진행해야 한다.

· Goal

이번 Design 보고서에서는 해당 프로젝트에서 사용한 클래스들을 Class Diagram을 통해 설명했으며, Use case에 대한 Sequence Diagram을 그려 동작 과정을 소개한다. 게임이 완성되었을 때의 StateMachine Diagram을 그려 어떤 상태가 될 수 있는지 보여준다. 해당 보고서를 읽고 나면 'Where rest is none'이 어떤 방식으로 Design 되었는지 알 수 있게 될 것이다.

2. Class diagram

· Class diagram



해당 클래스 다이어그램은 게임 Scene의 클래스들을 표현한 다이어그램이다. (실제 구현에는 더 많은 클래스가 사용되었으며, UI와 관련된 클래스는 현재 클래스 다이어그램에서 제외했다.)

클래스 다이어그램에 포함된 대부분의 클래스가 Monobehaviour라는 클래스를 상속한다. Monobehaviour 클래스는 Unity에서 스크립트를 작성할 때 사용하는 기본 클래스로, 모든 유니티 스크립트의 기반이자 게임 오브젝트의 동작과 상호작용을 제어하는 데 사용된다. 또한 코루틴을 지원하고 있다.

다음 페이지의 표는 클래스 다이어그램에 대해 연관된 게임 오브젝트를 나타내는 중요 클래스들을 선별하여 함께 설명했다.

Player	
InputManager	플레이어로부터 입력을 받아 다른 클래스로 입력값을 전달하는 역할을 담당하는 클래스다. 해당 클래스의 주요 역할은 입력 장치(키보드, 마우스)에 대한 입력 처리를 수행하는 것이다.
PlayerMove	InputManager 클래스로부터 받은 입력값을 통해 플레이어를 이동시키는 역할을 담당하는 클래스다. - 해당 클래스에서 사용되는 변수는 이동 방향, 이동 속도 등에 관련되어 있다. + 해당 클래스에서 사용되는 함수는 기본 이동, 달리기, 숨기 동작을 지원한다.
PlayerState	플레이어의 상태와 관련된 정보를 관리하며, 현재 체력 상태를 추적하고, 상태 이상 효과를 적용하는 역할을 담당하는 클래스다.

Enemy	
BattleEvent	적 오브젝트가 플레이어를 마주했을 경우 발생하는 전투 이벤트를 관리하는 클래스다. 해당 클래스는 전투 시작, 전투 종료 동작을 지원한다.
EnemyMove	적 오브젝트의 이동을 관리하는 역할을 담당하는 클래스다. 이동 경로 탐색, 타겟(플레이어)를 추적하여 이동하는 동작을 지원한다.
EliteEnemy	일반 적 오브젝트보다 더 강력한 적 오브젝트를 구현하는 클래스다.

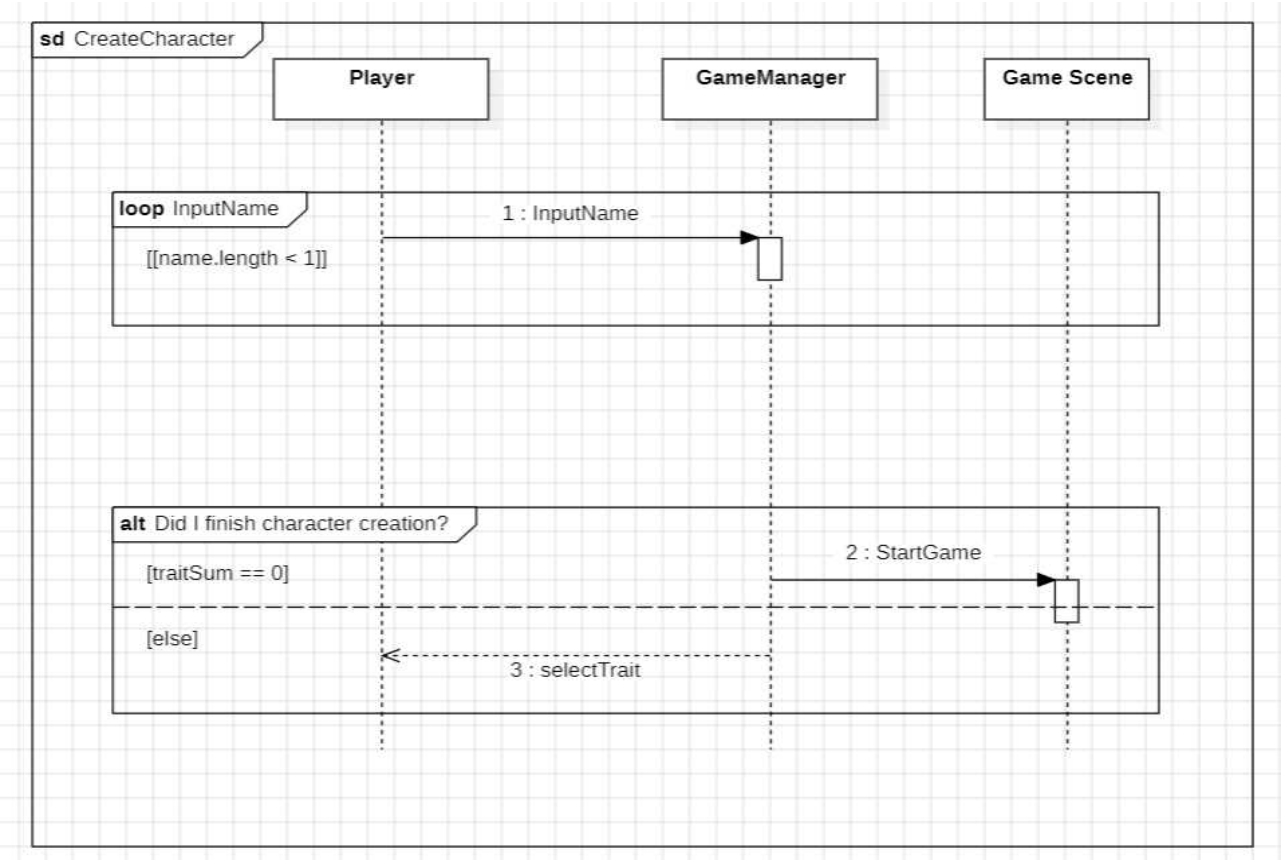
공통	
Health	게임에서 플레이어의 캐릭터, 적 오브젝트의 체력을 관리하는 역할을 담당하는 클래스다. 각 게임 오브젝트의 최대 체력, 현재 체력이 변수로 사용된다. 해당 클래스에서는 피해 (damage) 처리, 회복 처리, 사망 처리 동작을 지원한다.
WeaponManager	게임에서 무기(총)에 관련된 기능을 관리하는 클래스다.
Shoot	캐릭터나 적 오브젝트가 총을 발사하여 공격하는 동작을 처리하는 역할을 담당하는 클래스다. 총알 속도, 총알 데미지, 총이 비어있는지 여부가 변수로 사용된다. 해당 클래스는 발사, 재장전 동작을 지원한다.

Item	
ItemSpawner	게임에서 아이템을 생성하고 배치하는 역할을 담당하는 클래스다. 설정된 범위 내에 무작위로 아이템이 배치된다.

GetItem	특정 아이템을 인벤토리로 가져오고 해당 아이템을 게임 화면에서 제거하는 역할을 담당하는 클래스다.
InventoryManager	플레이어의 인벤토리를 관리하고 조작하는 역할을 담당하는 클래스다. 플레이어가 소지한 아이템을 인벤토리를 통해 관리하며 아이템 추가, 아이템 사용 동작을 처리한다. (해당 클래스는 상호작용할 수 있는 인터페이스를 제공한다.)

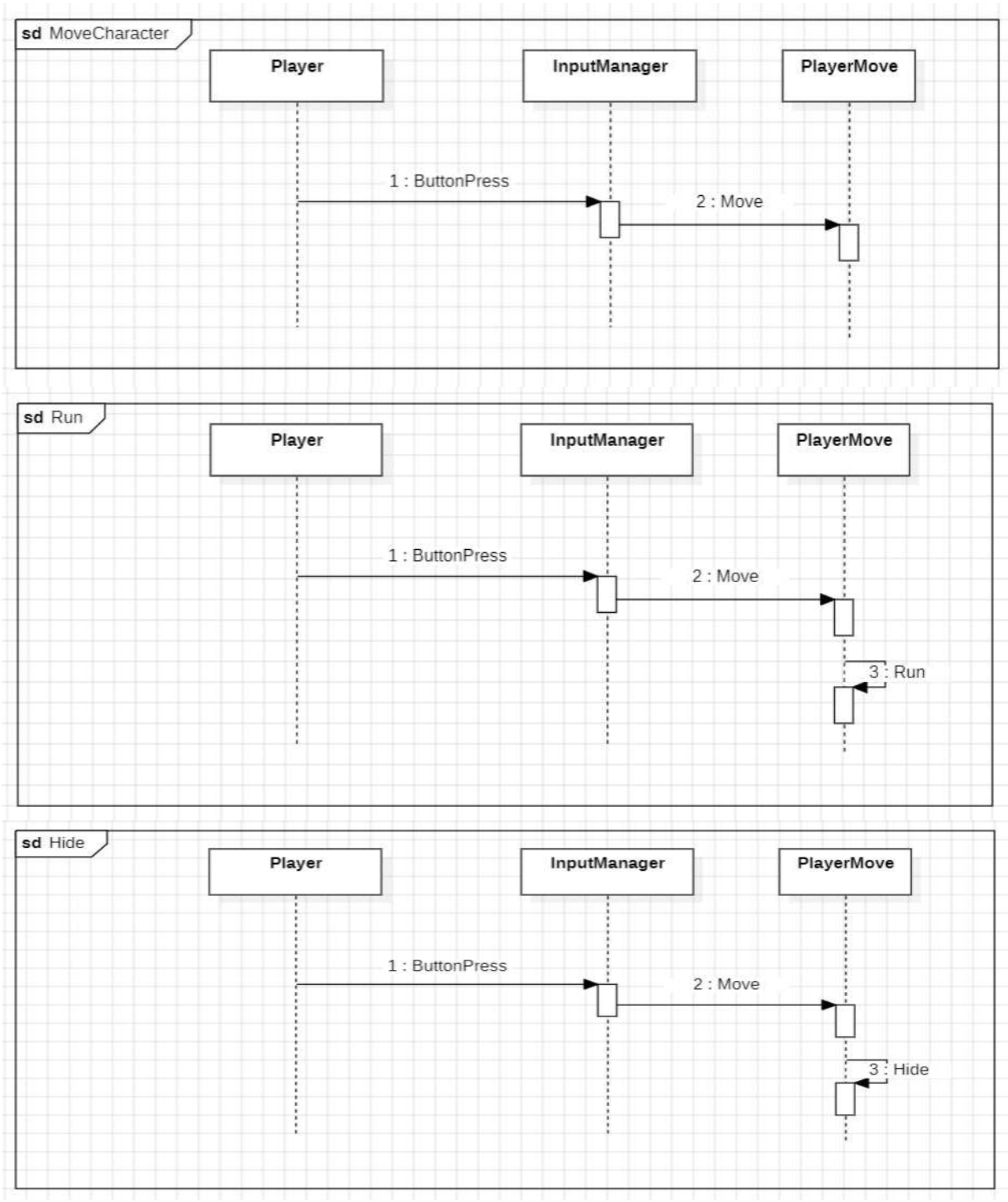
3. Sequence diagram

1) Create Character



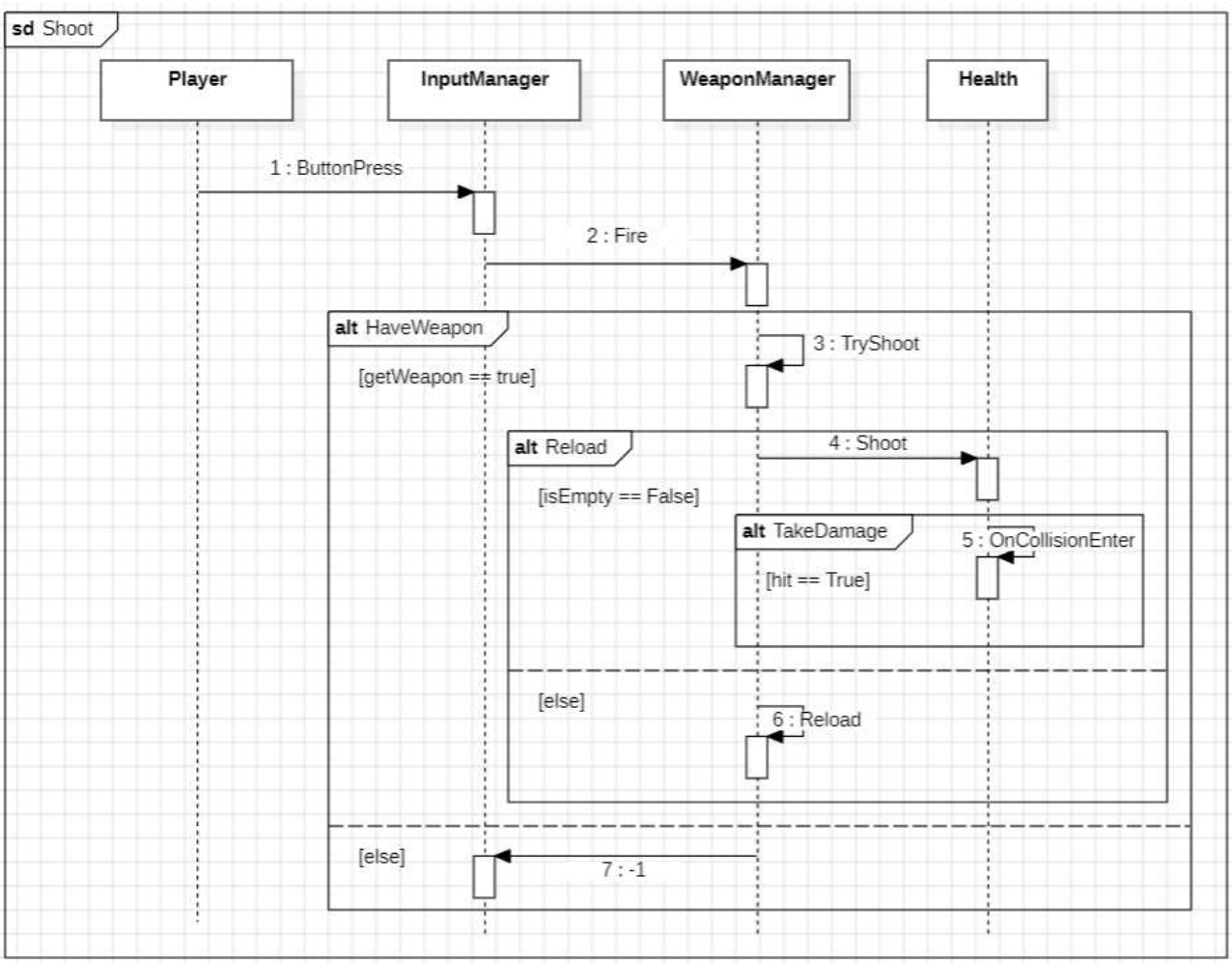
플레이어가 게임에서 사용할 캐릭터를 생성하는 Use case 'CreateCharacter'에 대한 Sequence diagram이다. 플레이어가 입력한 캐릭터 이름의 글자 수가 0인 경우 다시 이름을 입력하게 한다. 이후 Trait(특성) 중 플러스/마이너스 Trait을 적절히 선택하여 Trait 합계가 0이 되면 게임을 시작하고, 그렇지 않으면 합계가 0으로 맞춰질 수 있도록 selectTrait 메시지를 reply했다.

2, 3, 4) Move Character, Run, Hide



플레이어의 이동, 동작과 관련된 Use case 'MoveCharacter'에 대한 Sequence Diagram이다. 플레이어가 버튼을 눌렀을 때 InputManager가 특정 행동을 하기 위한 지정 버튼을 눌렀는지 판단하여 입력대로 처리한다. InputManager에서 처리된 행동은 PlayerMove에서 수행된다.

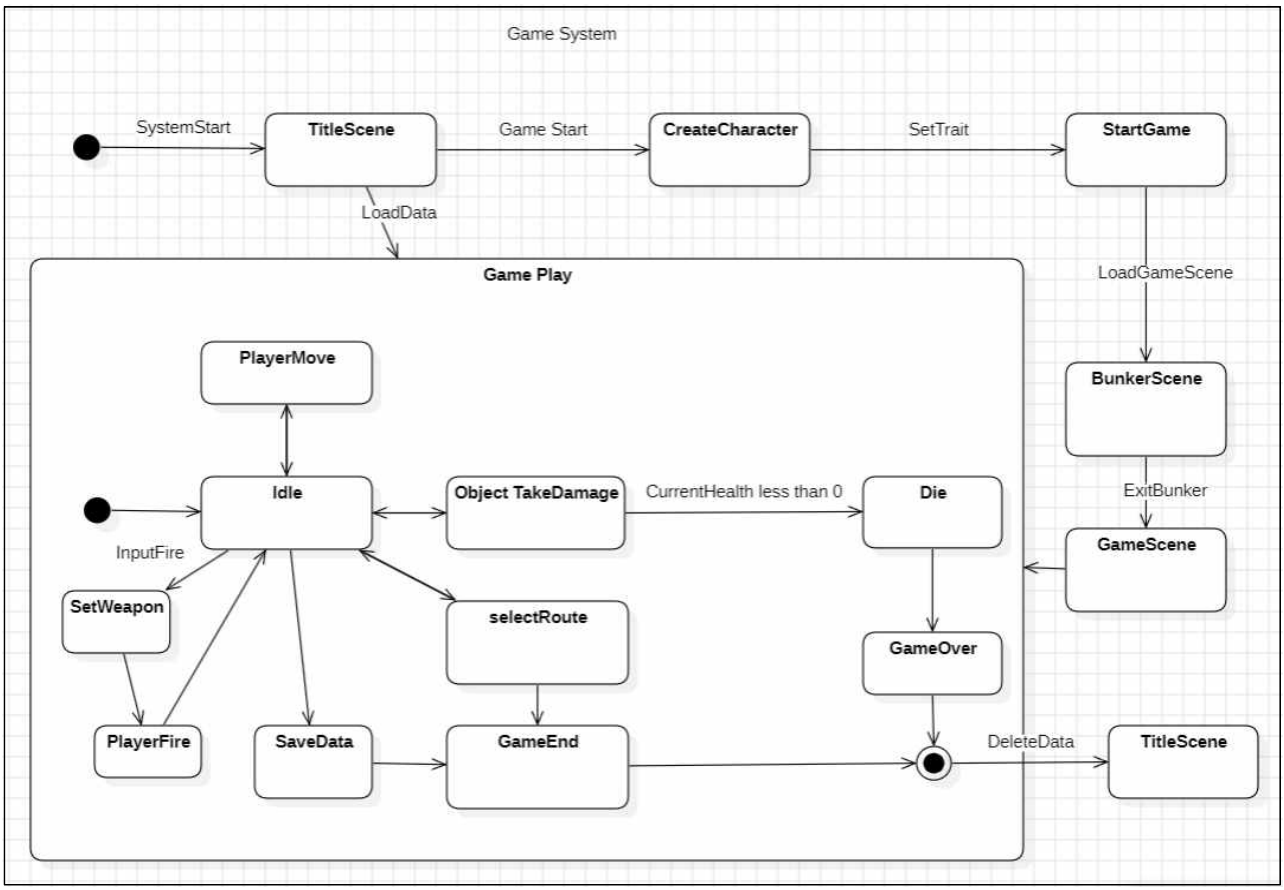
5) Shoot



플레이어가 무기를 발사하는 Use case 'Shoot'에 대한 Sequence Diagram이다. 플레이어가 무기를 발사하기 위해서 버튼을 누르면 InputManager에서 감지하여 WeaponManager로 발사하라는 Fire 신호를 보낸다. WeaponManager에서는 현재 무기를 소지하고 있는지 여부를 확인하고, 소지 중이라면 무기가 비어있는지 (발사 가능한 상태인지) 확인하고 비어있지 않으면 발사한다. 무기가 비어있다면 재장전을 한다. 총알이 Health 속성을 가진 게임 오브젝트와 충돌한 경우, Health 클래스에서 TakeDamage 메소드가 실행된다.

4. State machine diagram

· State machine diagram



플레이어가 시작 화면 (= Title Scene)에서 Game Start를 선택하여 게임을 시작하면 플레이어가 사용할 캐릭터를 생성하는 화면이 나온다. 캐릭터의 이름, 성별, trait (특성)을 설정한 다음 Finish 버튼을 클릭하면, 게임 스토리에 따라 플레이어는 벙커 내부에서 게임을 시작하게 된다. 그로 인해 캐릭터 생성 후 Bunker Scene으로 이동하게 된다. (만약 튜토리얼을 진행하고자 한다면 시작 화면에서 Tutorial을 선택하면 된다.)

Bunker Scene으로 이동한 플레이어는 키보드 입력을 통해 자신의 캐릭터를 이동시킬 수 있으며, 마우스 입력을 통해 카메라 회전을 하여 시야를 조정할 수 있다. 벙커에서 나가기를 선택하면 Game Scene으로 이동하게 된다.

총알에 맞게 되면 출혈 상태가 되어 체력이 감소하며, 체력이 0이 되면 플레이어는 사망하게 되어 게임 오버된다. 게임 오버가 된 경우 플레이어는 시작 화면 (= Title Scene)으로 이동하게 되며, 게임 진행 상황은 자동으로 말소되어 다시 게임을 시작해야 한다. (단, 게임 진행 도중에 진행 상황을 저장하면 게임 종료 후 시작 화면에서 Game Start를 선택했을 때 저장 내역을 load하여 이어서 플레이가 가능하다.)

Where rest is none은 다중 엔딩 게임으로 플레이에 따라 엔딩이 달라질 수 있으며, 각 엔딩 조건을 달성했을 때 해당 엔딩을 보고 게임이 끝이 나게 된다.

5. Implementation requirements

· H/W Platform requirements
- CPU : Intel i5+
- RAM : 최소 8GB (권장 16GB+)
- HDD / SSD : 10GB+

· S/W Platform requirements
- OS : Windows 7+
- Implement Language : C# (Unity)

6. Glossary

용어	설명
Sequence Diagram	객체 간의 동적인 상호작용을 시간 순서에 따라 모델링하여 표현한 다이어그램
StateMachine Diagram	객체의 상태와 상태 간의 전이 (transition)를 모델링하여 표현한 다이어그램

7. References

1) Structural Modeling, Behavior Modeling [강의자료]

2) Unity Documentation
https://docs.unity3d.com/kr/2023.2/Manual/Glossary.html