



# MoonSlayer (해외 게임잼)

☰ FrameWork	
☰ category	2D 게임
📅 develop period	@2020년 11월 1일 → 2020년 11월 30일
🔗 link	<a href="https://itch.io/jam/game-off-2020/rate/837025">https://itch.io/jam/game-off-2020/rate/837025</a>
☰ roll	프로그래밍
▼ size	Large
☰ 언어	C# Unity

개요

개발

본인 파트

2020.08.28회고

새로 알게된점

아쉬운점

의문점

## 개요

- Game Off 2020 에 출품함
- 7인 1개조 팀
  - 프로그래밍 4(애니메이팅 포함), 디자이너 2(사운드 그래픽 포함), 기획 1




- 플레이 영상

<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/b0535430-ec18-47a9-9a69-0753e7d3a542/PlayVideo.mp4>

보스 몬스터 (2:15~)

- 게임잼 링크 (깃허브 링크 포함)

Moon Slasher by Kwon770 for Game Off 2020  
Slash the enemies, jump up to 'The MOON' and meet 'The MOON' at the top. Available for Windows, macOS

 <https://itch.io/jam/game-off-2020/rate/837025>



- 플레이(다운로드) 링크

## Moon Slasher

Version 1.0.3 now available! ===== Story  
An unknown being from outer space has given intelligence and magic to the all the planets except Earth. Since then, the MAD

 <https://kwon770.itch.io/moon-slasher>



## 개발

▼ 우측의 삼각형을 누르게 되면 세부 사항을 볼 수 있습니다.

아래의 여러 항목에서 이 삼각형을 누르면 세부 사항을 볼 수 있습니다.

- git Flow를 사용하여 여러 브랜치를 통해 개발하고 PR과 코드리뷰를 통해 서로의 코드를 봄
- 게임을 출시한 이후에는 Issue를 통하여 버그 및 수정사항을 피드백함
- 이전에는 Sprite를 통해 완성된 애니메이션을 프레임 단위로 배치했다면 이번에는 유니티의 내부 기능을 통해 PSB파일을 이용하여 스켈레톤 애니메이션을 제작함

## 본인 파트

- 상속 특히 abstract, virtual적극 활용하여 했음
- Design Patterns
  - State machine  
보스가 여러 상태가 있고 해당 상태에서 각각의 특징이 있기 때문에 State machine 디자인 패턴을 사용함.
  - 싱글톤  
싱글톤을 활용하여 게임 매니저들의 호출 및 구현을 용의하게 함
- 애니메이팅  
각각의 몬스터와 보스의 PSB 파일을 받아 유니티 내부 기능을 통해 뼈대를 만들고 스켈레톤 애니메이션을 직접 제작함
- 몬스터 개발 파트를 맡음  
Enemy네임 스페이스를 사용하여 겹치는 일을 줄임
  - ▼ 몬스터 4종 (토끼, 늑대, 마스피플, 곰뱅이)제작

- 몬스터 구현
  - 행위 하나당 스크립트를 하나씩 제작하여 조합하는 형식을 채용하여 제작함
  - 1. Raser
    - 플레이어간의 거리를 계산하여 특정 거리 안에 들어올 경우 레이저를 발사함
    - 토끼면 플레이어가 토끼보다 높아질 경우 발사하지 않도록 설정
    - 총알을 쏘 경우 총 쏘는 파티클 출력
  - 2. Bullet
    - 총알이 발사될 시점의 플레이어 위치를 계산하여 저장함
    - 플레이어 위치를 기반으로 총알이 플레이어를 바라보게 해 줌
    - 플레이어에 맞게 될 경우 데미지를 주는 함수 실행
    - 땅에 맞게 될 경우 파괴
  - 3. Life
    - 죽을 경우 잘리는 이팩트를 발생하고, 자신을 비활성화하는 스크립트, 플레이어에서 슬래쉬할때 몬스터를 슬래쉬 성공할 경우 이를 호출함
  - 4. Pattern
    - 플레이어가 패턴 거리 안에 들어오면 패턴을 실행
    - 패턴은 코루틴을 이용하여 클타임을 관리하고 애니메이션 및 패턴을 실행하도록 함
    - 이때 패턴과 애니메이션은 virtual 로 선언하여 상속하는 오브젝트 들이 직접 구현하도록 함.
    - a. MeleeAttackPaatern
      - 근접 공격 패턴으로 늑대가 근접 공격을 하게 사용함.
      - 애니메이션을 override하여 늑대가 공격하는 모션을 취하고 소리를나게 함
      - 플레이어 위치를 계산하여 공격 범위 안에 있으면 데미지를 주는 함수를 실행함
    - b. ExplosionkPattern
      - 패턴을 override하여 폭발 할 경우 범위 안에 있으면 데미지를 줌
  - 5. Move
    - EnemyMove 스크립트를 만들어서 하위 오브젝트가 상속하여 기본 기능을 이용하게 함
    - a. EnemyMove(부모 스크립트)
      - 액티브 될 경우 지정된 위치로 이동함

Vector3.Lerp를 이용하여 이동하는 효과를 줌

대상(플레이어, 빈 오브젝트)와의 거리를 계산해서 걸리는 시간을 계산함

⇒ Lerp는 거리를 0,1로 정규화하여 이동하는데 이는 거리가 얼마나 멀던 같은 시간 내에 이동하게함 그러기 때문에 거리와 소요 시간을 측정하여 이동시킴

플레이어를 따라가는 함수를 virtual 빈 함수로 정의함

좌우를 보는 함수를 virtual로 만들고 채워넣음

b. 굼뱅이 Move

굼뱅이는 상하좌우로 볼 수 있기 때문에 좌우를 보는 함수를 override하여 상하도 추가함

c. 마스피플 Move

플레이어를 따라가기 때문에 해당 함수를 override하여 정의함  
플레이어의 위치에 따라 좌우를 볼 수 있게 만듦

d. 늑대 Move

늑대는 움직이지 않으므로 좌우만 보는 스크립트로 override함

▼ 몬스터 보기

공통 기능인 경우 상위 스크립트에서 구현하여 하위 스크립트에서 사용함(상속 활용)

• 토끼

- 플레이어가 감지 범위 내에 들어올때 바라봄
- 특정 위치를 왕복함
- 감지 범위안에 오면 총을 쏴.

• 늑대

- 플레이어가 감지 범위 내에 들어올때 바라봄
- 움직이지 않음
- 감지 범위 안에 오면 총을 쏴
- 플레이어가 감지 범위 내에 들어올때 할퀴기를 함(근접 공격)

- 굼벵이
  - 특정위치를 순회함
  - 플레이어가 다가오면 플레이어는 사망
  
- 마스피플
  - 네개의 좌표를 왕복함
  - 플레이어가 다가오면 쫓아감
  - 플레이어가 범위 안에 들어오면 3초뒤에 폭발하며 주변에 데미지줌
    - ⇒ 폭발을 하면서 몬스터의 색을 변경시키고, 몸체를 키우며 다른 다리들을 줄여주면서 폭발할거 같은 느낌을 줌
  
- 보스 몬스터 제작(14:49~)

보스 몬스터의 떨리는 연출, 회전등은 정적인 PSD파일을 받고 유니티의 애니메이션터를 이용하여 그림

FSM을 활용하여 몬스터의 패턴을 반복적으로 실행하게 함.

#### ▼ Moon 구현

##### 1. 보스의 채력을 관리하는 스크립트

보스가 타격 가능한지 상태를 가지고 있음

타격 가능할 경우 타격할때 체력을 보고 채력이 0이하면 죽는 애니메이션을 출력하고 죽는 소리를 냄

불가능 할 경우 바로 return false함

##### 2. 패턴을 컨트롤 하는 스크립트 생성

a. 등장패턴을 추가하여 뒤로 돈 상태에서 시작하게 하고, FSM을 실행하여 상태를 부여해줌

b. 패턴을 한번씩 실행해야 같은 패턴이 나오도록 함

(모든 패턴을 실행해야 중복 패턴이 나옴 = 1사이클 뒤에 동일한 패턴을 볼 수 있음)

n개중 랜덤으로 패턴을 뽑아냄

c. 2번째 패턴 사이클에서 마지막 패턴 과 동일한 패턴이 나오지 않게 함

ex) 5개의 패턴중 레이저 패턴이 마지막이면 다음 사이클 첫번째는 레이저 패턴이 나오지 않음

- d. Patten을 상속한 스크립트List를 통하여 해당 패턴의 상태로 가는 SFM을 만들어서 상태를 하나씩 지니고 있게하고, 2가지 패턴을 실행한 뒤에 때릴 수 있는 **탈진 패턴**을 실행

### 3. pattern

상위 오브젝트인 Pattern을 만들고 abstract를 사용하여 하위 스크립트에서 Play(애니메이션 출력 함수),Run(실제 패턴 함수)을 구현하도록 강제함  
⇒ 애니메이션은 패턴마다 다르기 때문에 하위 오브젝트에서 구현하기 위해 Play를 추가하고 각각의 애니메이션을 재생함

아래는 각각의 스킬의 핵심을 설명함

#### ▼ skill

##### 1. HatPattern

모자는 던지는 패턴, 달이 한바퀴 돌고 머리에 달린 모자가 사라지는 연출을 통해 모자를 던진것으로 보이게함

- 모자는 플레이어 위치를 받아서 위에서 아래로 떨어지고 부메랑 처럼 다시 플레이어 아래에서 올라오게함

##### 2. MadPattern

화내는 패턴, 달이 부들부들 떨면서 운석을 소환한다

- 플레이어 위치를 받아와서 운석을 액티브 하고, 달의 높이에서 소환한다

⇒ 오브젝트 풀의 기법을 활용함

이를 통해 필요없는 운석을 제거하거나 생성하는 소요를 줄이고, 운석을 재사용 가능하게 만듦

##### 3. SingPattern

달이 노래하는 패턴, 달이 노래하면 주변에 몬스터가 소환됨

- 특정 위치에 몬스터를 미리 생성하고 이 몬스터를 액티브해서 몬스터가 소환된 연출을 생성함
- 루프를 돌면서 몬스터가 이미 있을 경우 다른 몬스터를 액티브 하게 했음

- 플레이어가 몬스터를 죽이지 않고 유지시키는 경우를 고려하여 루프 도는 카운트를 세고 특정 카운트를 넘어갈 경우 루프를 빠져나오게함

#### 4. SlashPattern

달 입에서 레이저가 나오고 이를 좌/우 에서 반대쪽 45도까지 그리는 패턴

- 애니메이션으로 처리를 대부분 했기 때문에 해당 부분에서는 레이저를 키고, 시간을 측정해서 특정 시간이 지날 경우 원래 회전각으로 돌아가는 함수를 만들어서 사용함

#### 5. ExhaustPatterns

특정 시간동안 지치는 애니메이션을 내보내고 이 시간동안 공격 가능한 상태로 만들어줌

- BossLife스크립트를 불러와서 해당 스크립트의 IsCanAttack의 값을 조절해줌

## 2020.08.28회고

### 새로 알게된점

#### 1. PSB파일을 통해 스켈레톤 애니메이션 제작

이전에는 디자이너를 찾거나 해당 애니메이션을 서칭을 통해 구해야 했다면 이번에는 디자이너에게 부담을 줄이는 방법인 PSB파일을 통한 스켈레톤 애니메이션을 제작하는 방법을 깨우침

#### 2. 상속을 적극적으로 활용하고 abstract, virtual를 실제로 적용해 보는 경험을 가짐

##### abstract

하위 오브젝트에게 강제로 구현하게 하는 것, 몸체가 없으며, 여러개의 파생클래스에서 공유할 공통적인 정의를 제공함

##### virtual

재정의가 가능하지만 필요에 따라 재정의가 가능하지만 필수적이지는 않다, 자체적으로 기능을 제공함

##### Interface



보통 여러 클래스에 공통적인 기능을 추가할 경우 사용함 abstract와 비슷하다.

멤버변수를 사용할 수 없음, 대신에 프로퍼티를 사용가능하다

▼ 게임을 만들면서 사용하지 않았기 때문에 예제 첨부

```
public interface Animal
{
    void Speak();

    string Name
    {
        get;
        set;
    }
}

class Dog : Animal
{
    private string name;

    public void Speak()
    {
        Console.WriteLine(name + "->멍멍!");
    }

    public string Name
    {
        get
        {
            return name;
        }
        set
        {
            name = value;
        }
    }
}

Dog temp = new Dog();
temp.Name = "흰둥이";
temp.Speak(); //흰둥이->멍멍!
```

3. 서로 코드리뷰를 하면서 직관적인 함수명과 변수명의 중요성을 실감함

여러사람이 같이 협업하다 보니 어떤 역할을 하는지 잘 모르는 경우가 많아 직관적인 변수, 함수명의 중요성을 실감하게됨

4. FSM 유항상태 기계

유한 상태 기계의 존재를 알았으며 이는 게임에서 여러 방면으로 활용됨을 알게되었다.

내가 한 파트에서는 보스에서만 적용 하였으나 차후에 게임을 만들게 된다면 플레이어 몬스터등 각각의 오브젝트에 적용하면 좋을것 같다.

## 아쉬운점

시간이 적어서 오브젝트 풀링을 이용하여 레이저를 생성하지 못한점이 안타깝다

⇒ 게임의 규모가 엄청 크지 않아서 부하가 심각하지 않았으나 해보는게 좋았을 것이라는 판단

잔몹들의 완성도가 낮은점이 아쉬웠다.

⇒ 보스에만 너무 치중하여 잔몹들의 디테일이 조금 부족했던것 같다.

여러 사람과 같이 작업한만큼 주석을 달아야 했는데 달지 못한점이 아쉽다

⇒ 다음 프로젝트는 XML주석을 통하여 변수, 함수에 대한 설명을 적고 클린코드를 참조하여 직관적인 이름을 설정하도록 해야함.

## 의문점

실제 게임회사에서 몬스터를 이동시킬때 transLate처럼 몬스터의 위치를 이동하는지 addForce처럼 몬스터에게 힘을 주어서 이동시키는지 궁금하다.

⇒ 물론 게임마다 다르겠지만 RPG게임에서 움직이는 방식이 궁금하다.