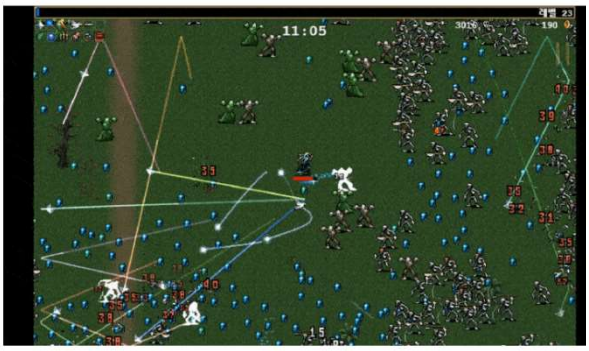
Portfolio : Eclipse

황민성

게임 소개

1. 게임 장르



모티브가 된 게임 “뱀파이어 서바이버즈”

포트폴리오 “Eclipse” 시연 영상

- 고전게임 “뱀파이어 서바이버즈”에 영감을 받은 로그라이크 슈팅 게임입니다.
- 천상과 지옥의 전쟁에서 플레이어의 선택에 따라 캐릭터의 외형이 변화하는 모습을 보여 줍니다.

2. 기술 스택

- Unity 2023.3.30f1, Rider 2024.1.2, GitLab

3. 개발 기간 및 개발인원

- 2024.09.18 ~ 2024.11.26 (약 2개월)
- 팀 프로젝트, 2인 제작

4. 담당파트

- 황민성(본인) : 메인 프레임워크, 무기 시스템, 데이터 관리, 몬스터 웨이브 등
- 이재욱(팀원) : UI 디자인, 아이템 드롭 시스템, 특수 웨이브, 맵 렌더링 등

5. 게임 영상 및 코드

- https://www.youtube.com/watch?v=WOJ5cdyHEBU
- https://github.com/Sicheu/Eclipse.git

사용 기술

1. 디자인 패턴

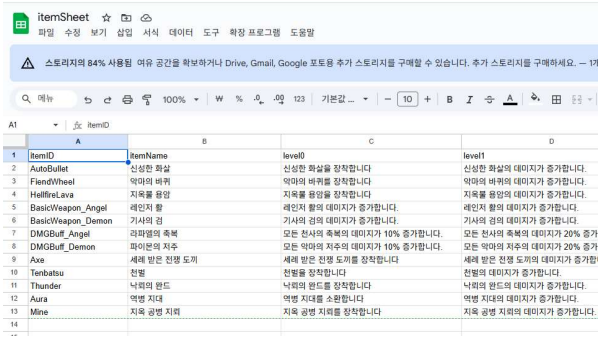
싱글턴 (Singleton)	다양한 매니저 클래스들에 사용되었으며 게임 전역에서 상태, 데이터, 리소스 등을 하나의 인스턴스로 관리하고 접근성을 확보하였습니다.
오브젝트 풀 (Object Pool)	무기, 이펙트, 몬스터 등 반복적으로 생성 및 삭제되는 오브젝트들을 풀링하여 성능을 최적화하였습니다. 또한, Scriptable Object와 연결하여 사용 편의성을 향상시켰습니다.
유한상태기계 (FSM)	캐릭터들의 현재 상태를 독립적으로 설계한 후 각 상태를 유기적으로 전환하여 코드 가독성 향상 및 유지보수의 용이함을 유도하였습니다.
전략 (Strategy)	무기 클래스 구조는 전략 패턴에 가깝게 확장성 중심 설계가 되어 있어, 모든 무기가 가져야 할 핵심 기능은 유지하면서 무기에 따라 다양한 공격 방식을 구현할 수 있었습니다.
옵저버 (Observer)	특정 이벤트(레벨업, 아이템 선택 등) 발생 시 관련 시스템들이 자동으로 반응하도록 설계하였습니다.

사용 기술

2. 응용 기술

[JSON 데이터 관리]

무기 및 아이템의 UI 텍스트를 구글 시트에서 관리하고 JSON으로 변환하여 유니티 프로젝트 내에 업로드하였습니다. 이를 통해 데이터 중심 개발을 구현하였으며, 유지보수와 밸런스 조정이 수월하였습니다.

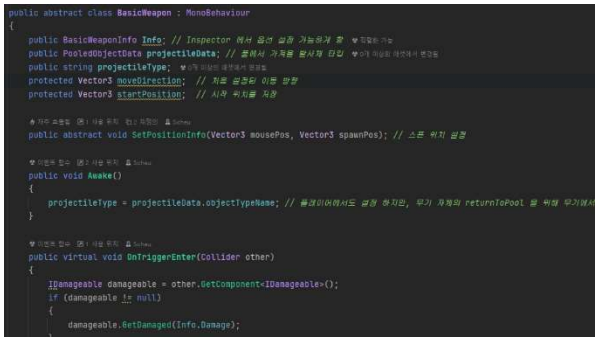


UI 값이 입력된 구글 시트

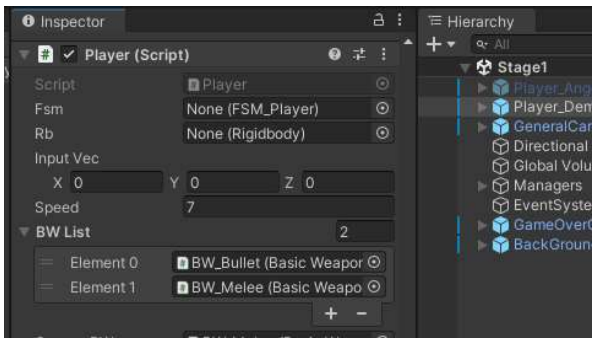
값을 받아 게임에 적용된 모습

[커스텀 에디터]

플레이어 및 보스 몬스터의 컴포넌트 구조가 복잡하여 특정 기능 변경이 어려워졌습니다. 이에 커스텀 에디터 기능을 활용하여 Inspector를 확장하였습니다. 이를 통해 필드 구성을 시각화하고 반복 입력을 자동화하여 개발 편의성과 정확도를 향상시켰습니다.



인스펙터에 표기 불가한 추상 클래스



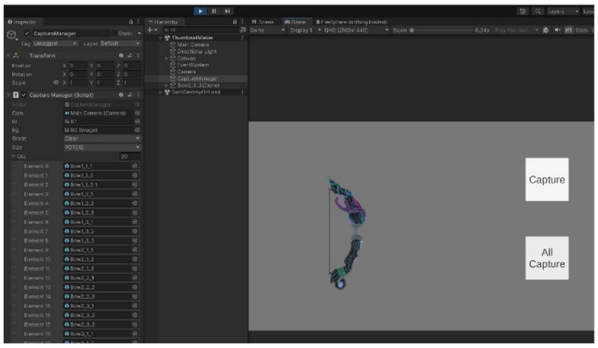
커스텀 에디터를 통해 인스펙터에 표기

사용 기술

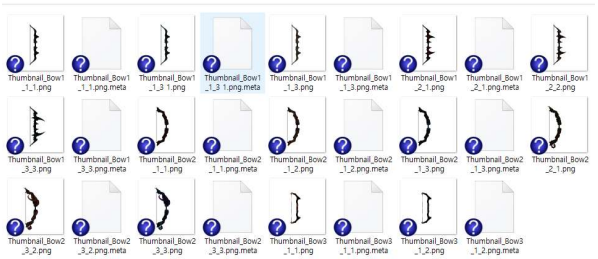
2. 응용 기술

[썸네일 메이커]

UI 요소 및 게임 오브젝트의 썸네일 이미지를 자동으로 생성하는 툴을 구현했습니다. RenderTexture 와 Texture.2D.ReadPixels()을 활용해 카메라 출력을 실시간으로 캡처하고, 코루틴을 통해 렌더 타이밍을 고려한 비동기 이미지 캡처를 안정적으로 처리하였습니다. 배경 색상 및 해상도를 설정할 수 있으며 썸네일을 생성할 오브젝트를 리스트에 등록하면 한 번에 대량의 썸네일을 생성할 수 있는 강력한 툴을 완성했습니다.



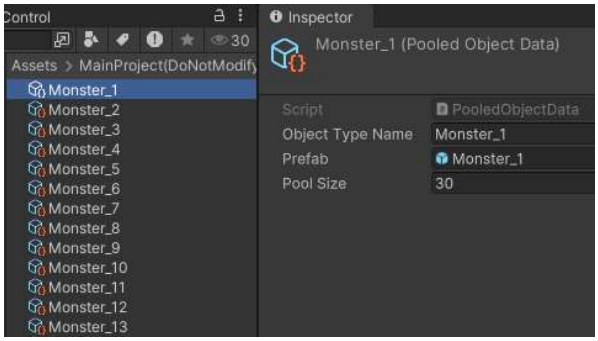
썸네일 메이커 동작 모습

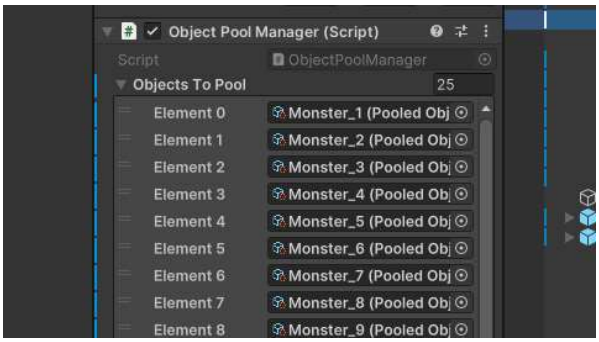


제작되어 로컬에 저장된 썸네일 파일들

[사용 편의성을 높인 오브젝트 풀]

게임 오브젝트 풀링을 단순 코드 기반이 아닌 ScriptableObject와 연동하여 구성하였습니다. 각 풀 오브젝트와 초기 수량, ID 등을 데이터 자산으로 관리할 수 있어 코드 수정 없이 풀 구성 변경이 가능하며 다양한 타입의 오브젝트를 유연하게 관리할 수 있도록 확장하였습니다. 이로 인해 코드를 설계한 본인뿐만 아니라 팀원들도 쉽게 사용할 수 있도록 설계함으로써 사용성과 협업 효율을 동시에 높였습니다.

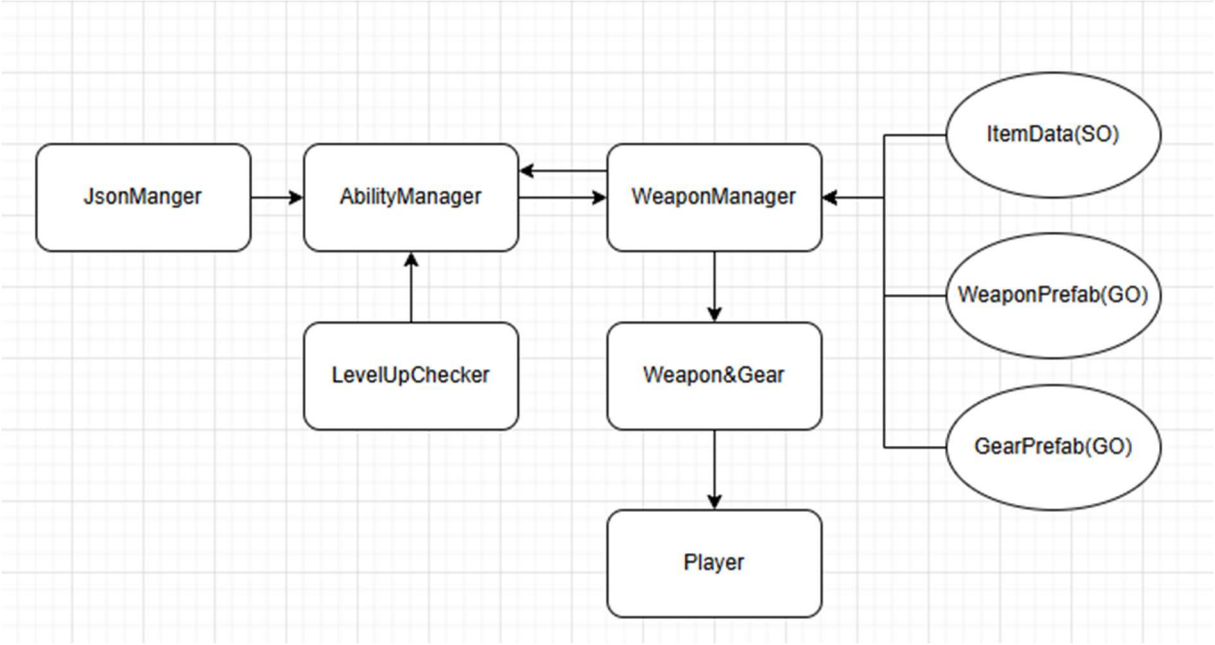




코드 수정 없이, 단순히 ScriptableObject를 만들어 오브젝트 풀에 등록하여 사용하는 모습

기능 구현

1. 무기 및 레벨업 시스템



[Weapon Manager]

- 무기와 기어의 프리팹과 그 데이터를 딕셔너리에 매핑하여 들고 있다가 외부 요청에 따라 이를 조합하여 실제 장비를 생성합니다.

- 레벨업 이벤트 발생 시 선택된 보상에 따라 변경된 아이템 데이터에 기반하여 기존 무기를 업그레이드하거나 새로운 무기를 생성합니다.

[Item Data]

- 각 무기 및 기어의 이름, 타입, 아이콘, 데미지, 발사 속도 등 장비의 기반이 되는 데이터를 Scriptable Object 로 구현하여 관리 및 유지보수를 가능하게 합니다.

[Json Manager]

- UI에 표시될 장비 선택지의 이름 및 레벨에 따른 기능 설명을 구글 시트로 만든 Json 파일에 기반하여 외부에 알려줍니다.

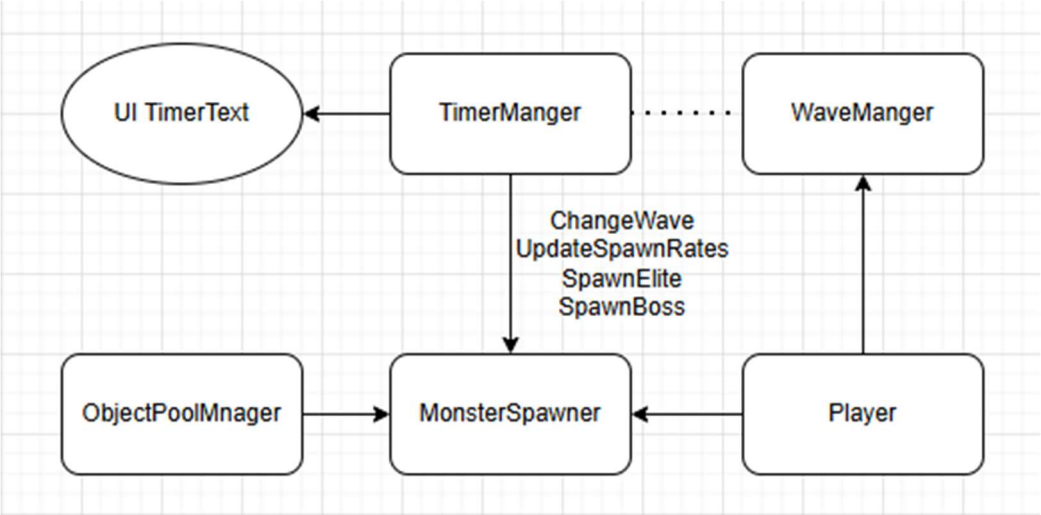
[Ability Manager]

- 레벨업 이벤트 발생 시 게임을 일시정지하고 Weapon Manager 로부터 매핑된 데이터를 받

아와 보상을 선택하는 패널을 생성합니다. 패널이 선택되면 Weapon Manager에 선택된 장비의 레벨업이 발생했음을 알립니다.

기능구현

2. 웨이브 및 타이머 시스템



[Timer Manager]

- 코루틴을 통해 프레임 기반으로 시간의 흐름을 감지합니다. 설계된 로직을 통해 흐른 시간에 따른 이벤트 발생을 제어합니다.

- 몬스터 스포너에서 스폰할 웨이브의 종류나 비율을 조정하며 엘리트 및 보스 몬스터의 타이밍을 제어합니다.

[Monster Spawner]

- 설정된 웨이브 정보에 따라 오브젝트풀에서 생성할 몬스터를 불러옵니다. 불러온 몬스터는 현재 플레이어의 위치 정보를 기반한 알고리즘에 의해 플레이어로부터 특정 거리 바깥에서 무작위 위치에 생성됩니다.

[Wave Manager]

- Timer Manager 와 직접적으로 연결되어 있진 않으나, 씬의 시간 흐름을 감지합니다.

- 시간의 흐름에 따라 주기적으로 생성되는 몬스터 웨이브가 아닌 특수한 이벤트 웨이브를 발생시킵니다.

회고록

[잘 된 점]

1. **확장성 중심의 설계 구현**
- 무기 시스템, 오브젝트 풀 등 핵심 기능을 다양한 기술을 활용하여 유연하게 설계함으로써 추후 유지보수와 확장성이 뛰어난 구조를 만들 수 있었습니다.

2. **협업 효율을 고려한 개발**
- 팀원과의 역할 분담이 명확했고, 에디터 커스터마이징, 데이터 시각화 등을 통해 협업 시의 진입 장벽을 줄이는데 기여했습니다.

3. **기능 중심의 도구 개발**
- 썸네일 메이커, 커스텀 오브젝트 풀 등 반복 작업을 자동화할 수 있는 유틸리티 도구를 직접 제작함으로써 개발 생산성을 실질적으로 향상시켰습니다.

[개선할 점]

1. **일정 계획표의 부재**
- 개발 초기에는 팀원들과 일정 계획표 공유했지만, 프로젝트 후반기에는 이를 성실히 관리하지 않고 구두 소통에 의존하게 되었습니다. 그로 인해 일부 작업이 누락되거나 지연되는 문제가 발생했고, 일정 관리 도구의 중요성을 다시금 실감하게 되었습니다.

2. **기획과 개발 간의 간극**
- 일부 시스템은 기획 당시 단순해 보였지만, 실제 구현 과정에서 기술적 난이도가 높아졌고, 그로 인해 팀원이 이탈하는 상황도 발생했습니다. 앞으로는 프로토타이핑 단계에서 기술적 타당성과 리스크를 보다 면밀히 검토할 필요가 있다고 느꼈습니다.

3. **가독성 중요성 인식**
- 협업을 위해 팀원이 사용할 수 있도록 여러 매니저 클래스와 개발 도구를 제작했지만, 실제로는 적극적으로 활용되지 않았습니다. 이 경험을 통해 남이 작성한 코드를 이해하는 데 얼마나 어려움이 있을 수 있는지를 체감했고, 가독성 높은 코드와 친절한 문서화의 중요성을 깊이 느끼게 되었습니다.