BANG

소프트웨어융합학과 김수연, 신동준, 최현승, 허건호

디지털콘텐츠학과 마현아, 이윤아

생용맹한소인들이여, 거인들의소굴에서 보물을 되찾아오라. ""

'Wi, Zard!' 는 **거인들이 마법사 소인의 마을을 공격**해 보물을 빼앗고, 두 소인이 이를 되찾기 위해 **거인의 마을**로 잠입하는 여정을 그린 게임이다. 유저는 소인 '**Wi'와 'Zard'의 시점**으로 게임을 플레이하게 된다.

퍼즐 소울라이크 협동 ---- 거인의 세계 소인의 여정

동료와 함께 **미지의 세계**를 추리하며 헤쳐나가자! 🤫

'Wi, Zard!'는 총 **세 개의 스테이지**로 구성되어 있으며, 스테이지당 **세 단계의 퍼즐**이 존재한다. 고유한 컨셉과 풀이 방식을 가진 각 스테이지를 플레이하며 참신한 재미 를 경험할 수 있도록 유도한다.

모든 퍼즐은 두 플레이어가 정보를 공유하고 문제를 해결해야 하는 '협동 요소'를 기반으로 한다. 앞서 사용된 퍼즐 기믹들은 퍼즐과 스테이지를 넘나들며 응용되어 게임의 개연성을 강화한다.



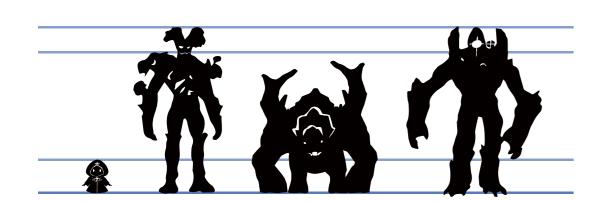






▶ 각 스테이지는 게임의 흐름, 보스의 컨셉에 맞춰 설계했다. 밝은 지상에서 시작해 점차 어둡고 깊은 지하로 내려가는 하강 구조를 띄고 있으며, 게임을 진행할수록 더욱 강력한 보스를 상대하게 되어 몰입감을 더한다.

겁 없이 나아가는 자, **큰 시련**을 마주할지니. ""



▶ 퍼즐을 클리어하면 레이드가 시작된다. 레이드의 보스는 거인이며, 소인과 대비되는 압도적인 크기로 긴장감과 스릴을 선사한다.

보스들은 **직전 스테이지의 퍼즐 기믹을 응용한 패턴**을 지니고 있어, 유저는 퍼즐의 기억과 함께 **협동하여 이를 파훼**하는 재미를 느낄 수 있다. 레이드가 끝나면 다음 스테이지로 넘어가 또 다른 퍼즐을 마주하게 된다.

+ PLUS 웅장한 오케스트라와 신비로운 선율이 선사하는 몰입감

'Wi, Zard!'는 작곡 프로그램을 이용해 **손수 제작한 배경 음악**으로 게임의 컨셉을 극대화한다. 소인 시점에서 거인을 바라보았을 때 느껴지는 감정과 **웅장한 분위기**를 표현하고자 **오케스트라 장르**를 채택했다.

퍼즐 스테이지

호기심을 자극하는 **윈드와 퍼커션, 긴장감**을 더하는 **반음**

보스 스테이지

웅장함을 극대화하는 **호른과 트럼펫, 저음역대**를 강조한 스트링 퍼즐 스테이지의 **멜로디 진행 및 BPM**을 활용하여 **연결성 강화**



* Ableton Live 12 Standard, BBC Symphony Orchestra 이용

보스 몬스터 구현 : Behavior Tree

✓ 상태 패턴의 필요성

동적 행동 관리

AI는 환경이나 내부 상태 변화에 따라 행동을 변경해야 하며, 상태 패턴은 이러한 동적 행동 변화를 효율적으로 관리한다.

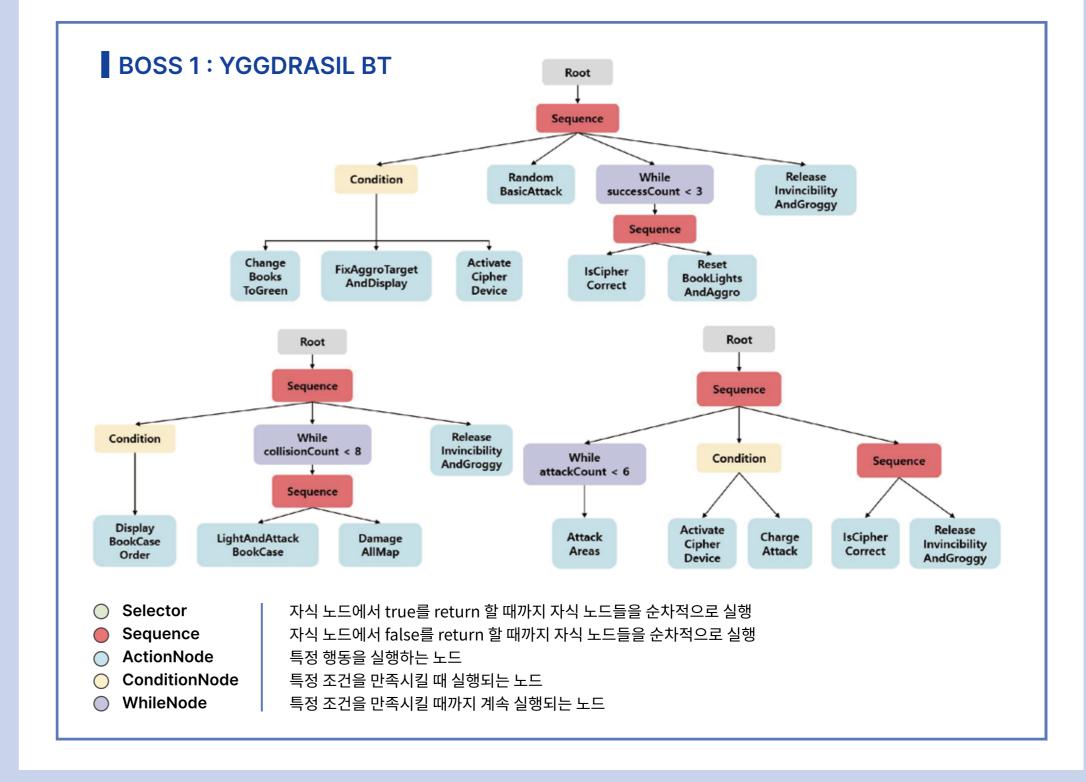
코드 모듈화 및 유지보수성 향상 각 상태를 별도의 모듈로 분리하여 코드의 가독성 및 재사용성을 높이고, 새로운 상태추가 시 기존 코드에 최소한의 영향만 준다.

✓ 상태 패턴: Finite State Machine(FSM)

간단한 상태 전이와 행동을 구현할 때는 직관적이고 효율적이지만, 행동이 복잡해질수록 필요한 상태와 전이의 수가 기하급수적으로 늘어나는 상태 폭발(State Explosion) 문제가 발생한다.

✓ 상태 패턴: Behavior Tree(BT)

레이드에 등장하는 보스의 행동은 복잡하므로, Finite State Machine을 사용하는 것은 무리가 있다고 판단하여 AI의 복잡한 행동 구현에 자주 사용되는 Behavior Tree를 사용하였다.



멀티 플레이 구현 : Photon Unity Networking (PUN)

• 갱신의 빈도에 따라 적절한 방법을 선택 (PhotonView -> RPC -> Custom Property), 빈도 수로 정렬

Tip 이벤트를 본인에게만 발생시킬 건지, 다른 플레이어에게도 발생시켜야 할지 명확하게 판단해야 한다.

✓ PhotonView

네트워크 오브젝트의 상태를 **자동으로 동기화**하는 컴포넌트

PhotonTransformView

PhotonAnimatorView

플레이어와 보스의 **위치, 회전, 크기** 동기화

플레이어와 보스의 **애니메이션** 동기화

관찰 가능한 컴포넌트 리스트에 'PhotonTransfromView'와 'PhotonAnimatorView' 컴포넌트를 할당시킴으로써 네트워크상에서 자동으로 **플레이어와 보스의 위치, 애니메이션**을 동기화한다.

플레이어의 행동은 모두 **PhotonView를 통해서 관찰된 후 동기화**된다. 즉, Wi가 이동했을 때 다른 세계(Zard의 컴퓨터)에 있는 Wi의 Clone은 PhotonView를 통해서 이동 명령을 받는다. 그러므로 동기화를 위해서 PhotonView가 필수적으로 플레이어의 컴포넌트로 있어야 한다.

✓ Remote Procedure Call (RPC)

특정 네트워킹 함수를 호출하여 그 결과를 다른 플레이어에게 동기화하는 방법 ex) 보스의 체력을 깎는 함수, 플레이어가 무기를 교체하는 함수 등에 사용

Custom Property

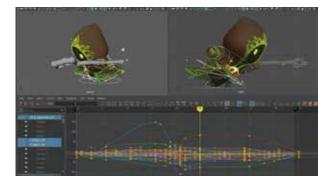
방이나 플레이어의 상태를 **Photon 서버에 key-value 쌍으로 저장하고 동기화**하는 방법 ex) 게임 시작 전, 준비 버튼을 통한 로컬 플레이어의 준비 여부를 동기화

'Wi, Zard!'의 아트워크 소개

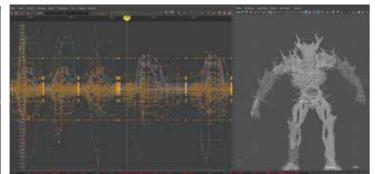
✓ Ingame Key 애니메이션 제작

자유로운 카메라 시점 이동이 가능한 Wi, Zard 특성 상 인게임 캐릭터들의 애니메이션은 플레이어의 몰입도와 밀접한 연관이 있다. 따라서 이를 인지하고 애니메이션 기획을 통해 체계적인 애니메이션 차트를 짠 다음, Maya를 사용하여 플레이어와 보스 캐릭터 Rigging 작업을 진행한 뒤 3D 애니메이션을 제작했다.

각 캐릭터별 성격과 특징을 반영하여 모션을 기획한 뒤 Idle, Walk, Jump, Groggy 등 기본 모션과 더불어 플레이어의 공격, 궁극기 모션, 보스의 각 패턴 별 모션 등 게임에 필요한 애니메이션을 작업했다.

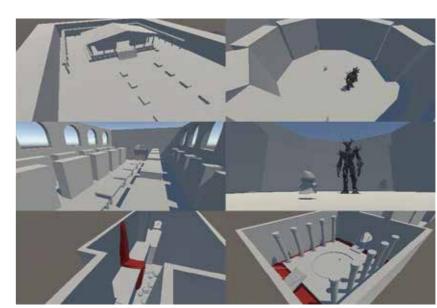


1) Player: Wi & Zard





2) Boss1 : Yggdrasil 3) Boss2 : Vanta





✓ 컷씬 애니메이션 제작

게임 플레이로 구현이 어려운 공간이동과 보스 등장 상황에 맞춰 **컷씬을 제작**했다. 캐릭터가 처한 **상황이나 감정 등을 연출**하여 몰입도와 긴장감을 높인다. 또한 이동 경로, 플레이 방식, 중요한 아이템 등을 직관적 으로 안내하여 게임 플레이의 이해를 돕는 것이 그 목적이다.

✓ 인게임 에셋 제작

캐릭터에게 필요한 아이템과 퍼즐 풀이에 사용하는 상호작용 오브젝트를 제작했다. 이를 위해 기획 단계에 서 스케치 및 더미 모델링을 통해 캐릭터—오브젝트 간 사이즈를 조절하여 체계적인 토대를 마련했다. 또한 플레이에 필요한 정보를 알려주는 퍼즐 단서, 약도 등에 대한 텍스쳐를 제작했다.