게임으로 소통하는 개발자

박상현 Park Sang Hyeon

연락처: 010-####-2778

이메일: sh010510@naver.com

Github: https://github.com/swatper

프로필

Education 학력사항

동의대학교 컴퓨터소프트웨어공학과 (2020.03 ~ 2026.02)

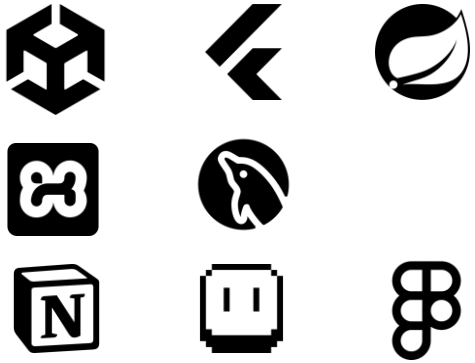
- 학점: 4.03/4.5 (전공: 4.13)

정보처리기사 (2025.12.24)

Skills 보유 역량

Languages

Tools



Activities 활동

2026년

- BIC 커넥터즈 6기 | (06.01 ~ ing)
- 2025학년도 DEU 디지털 마스터 수료 | 동의대학교 (01.20)

2025년

- 2025 동의 AI.SW 융합 해커톤 대회 특별상 | 동의대학교 (07.10)
- 오렌지 플레닛 게임 유저단 활동 | 스마일게이트 (07.04)

2024년

- 2024년 학생 창업유망팀 300+ 경진대회 예비트랙 수료 | 교육부 (08.31)
- 2024 부산 진구 연합 창업 교육 수료 및 아이디어 경진대회 장려상 | 동의대 Link (05.10)
- 붕어빵 유니버스 공모전 1차심사 선정 | 컴투스 (04.15)
- 2024학년도 창업 동아리 활동 Insight | 동의대학교

2023년

- 2023 통합 성과 경진대회 우수상 | 동의대학교 (11.16)
- 2023학년도 창업 동아리 활동 Insight | 동의대학교

Project 01

My Profile Unity

통합 게임 포트폴리오 플랫폼

프로젝트 개요

기간: 2개월(~ing)
인원: 1명

프로젝트 소개

WebGL기반 라이브 기술 아카이브 인터렉티브 마을(Debug Village)와 뱀서라이크(Dev Survival)을 결합한 포트폴리오

지속적 유지보수 및 기능 추가를 위한 확장형 설계로 견고한 소프트웨어 역량 증명

TECH STACK

Unity URP HTML/CSS JavaScript API

주요 기능

웹 플렛폼 제어 및 네이티브 기능 확장

WebGL 빌드 환경 한계를 극복하기 위해 jslib 기반 브릿지를 구축하여 브라우저 전체 화면 상태 및 클립보드 기능 연동

성능 최적화 및 리소스 관리

Queue 기반 Object pooling 및 Sprite sheet 배칭을 통해 대규모 개체 런타임 오버헤드 최소화

Github Actions와 Gist를 활용한 데이터 관리 자동화

Github Actions 및 Python 크롤링을 활용하여 외부 데이터를 Gist에 가공 배포하는 자동화 파이프라인 설계

KEY ACHIEVEMENT

지속적인 기술 고도화와 최적화를 추구하는 게임

- 크로스 플렛폼 최적화: 웹-엔진 제어를 통해 WebGL 환경의 자원 및 환경적 제약 해결
- 안정적 런타임 확보: 브라우저 자원 제한 극복 및 리소스 관리를 통해 안정적인 프레임 확보
- 엔지니어링 자동화: 데이터 수집부터 배포까지 자동으로 갱신되는 자동화 파이프라인 구축

상세

Notion Portfolio Link

웹 플렛폼 제어 및 네이티브 기능 확장

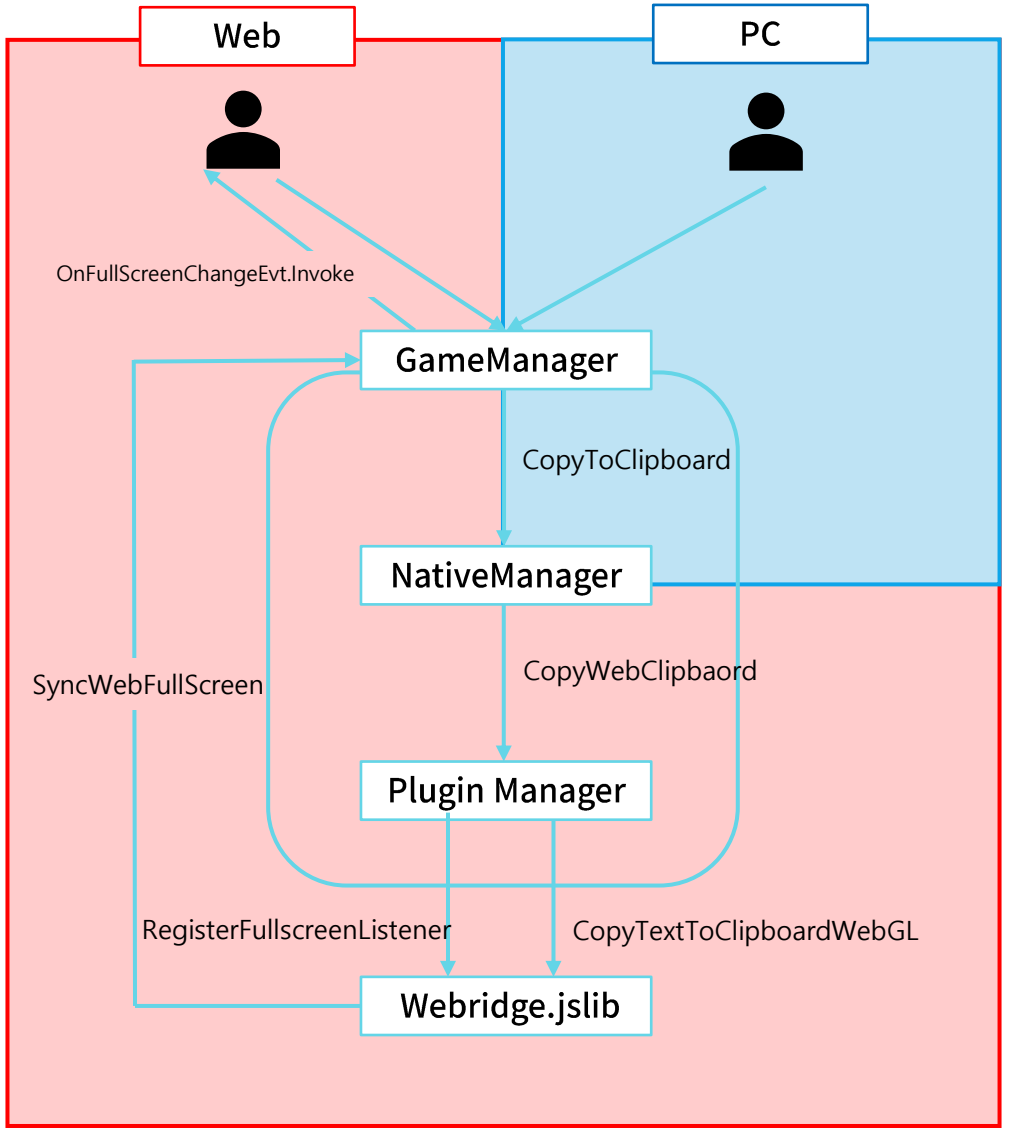
DOM ↔ 엔진 간 자원 경합 최적화

포커스 이벤트 활용으로, 배경 DOM 애니메이션을 일시 정지시켜 자원 경합 최적화

jslib 플러그인을 활용한 웹 네이티브 기능 확장

브라우저 API 호출을 통해 웹 환경의 클립보드 복사 기능 제공

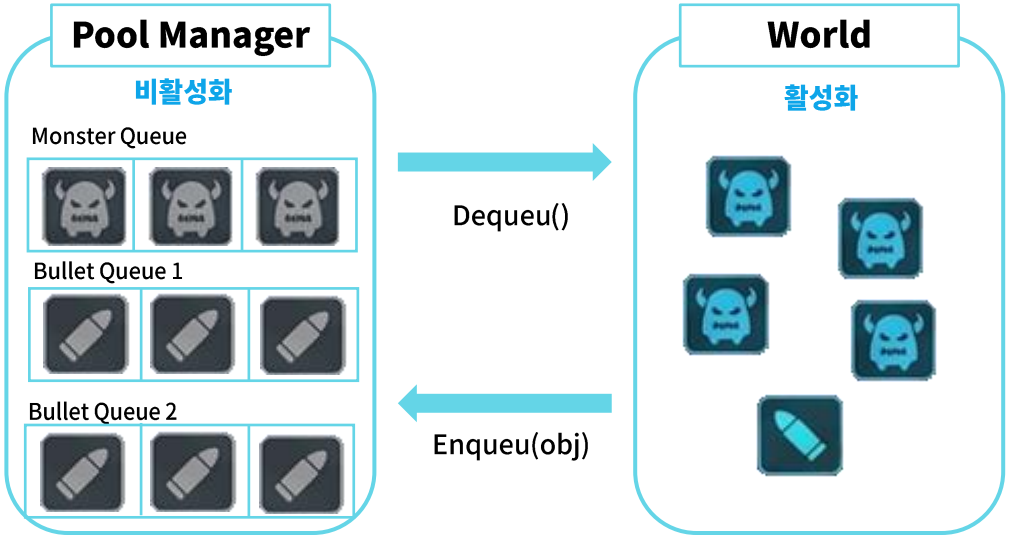
JavaScript 이벤트 리스너를 연동하여 실시간 웹 전체화면 상태 동기화



성능 최적화 및 리소스 관리

독립적 Object Pooling 시스템

뱀서 라이크 씬 내의 수십개의 몬스터와 투사체 관리를 위한 Queue 기반 자체 풀 구축



Draw Call 최적화

Muiti Sprite 모드로 하나의 에셋에 필요한 영역만 참조
에셋 시트화를 통한 Draw Call 감소로 렌더링 부담 감소

Scriptable Object 활용

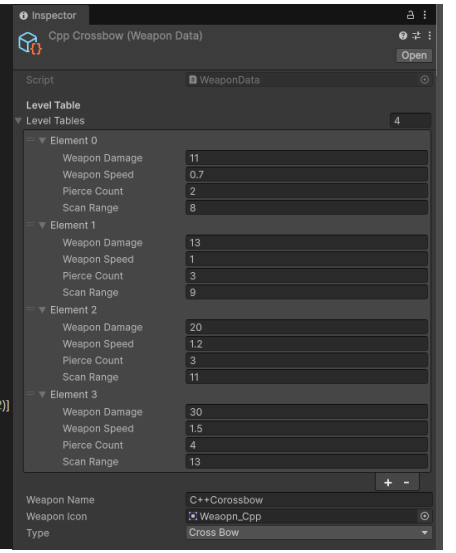
제네릭 구조체 기반 클래스로 확장성 확보
중심 설계를 통해 메모리 참조 효율 증가 및 레벨 데이터 확장성 확보

```
using Core.Define;


/// <summary>
/// Weapon Struct
/// </summary>
[System.Serializable]
참조 6개
public struct WeaponStat
{
    public float WeaponDamage;
    [Tooltip("초당 공격 속도")]
    public float WeaponSpeed;
    public int PierceCount;
    [Tooltip("Radius")]
    public float ScanRange;
}



/// <summary>
/// Weapon Scriptable Object
/// </summary>
[CreateAssetMenu(fileName = "Weapon", menuName = "ScriptableObject/WeaopnData", order = 2)]
Unity 스크립트 참조 2개
public class WeaponData : BaseUpgradeData<WeaponStat>
{
    public string WeaponName;
    public Sprite WeaponIcon;
    public UpgradeType Type;
}
```



Github Actions와 Gist를 활용한 데이터 관리 자동화

정기적 데이터 크롤링

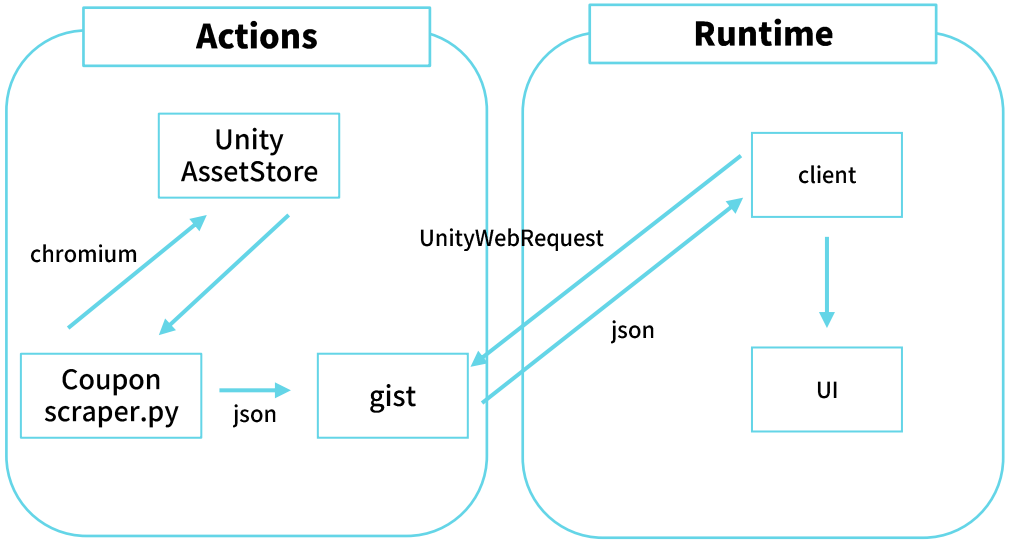
Github Actios 워크플로우(yaml)를 통한 주기적인 크롤링 자동화
Python 크롤링 기반의 Unity 무료 에셋 정보 수집 구축

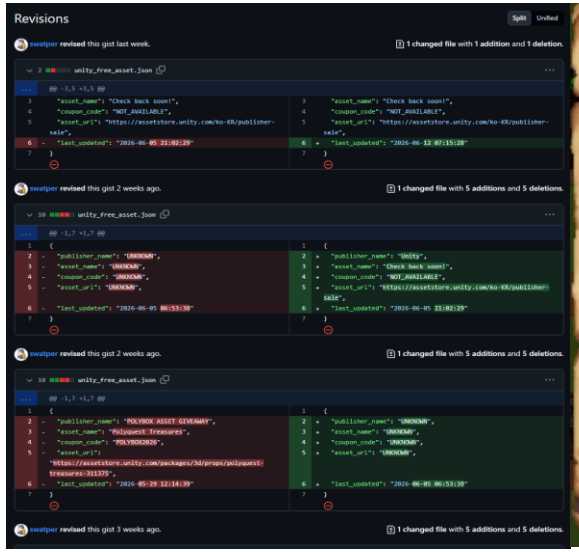
클라우드 데이터 관리

수집된 정보를 Json 형태로 Github Gist에 배포

인게임 반영

게임 실행 시 Gist 데이터 수집 및 인게임 UI에 반영





이슈 해결 - 웹 환경의 제약

환경에 기능 제약

Environment

브라우저 샌드박스 보안 및 환경적 API 접근 제약

유니티 내장 클립보드 API가 WebGL 빌드 환경에서
작동하지 않아 기능이 제한되고, 브라우저 자체
전체화면 전환 시 인게임 상태와 동기화 되지 않는 문제 발생

↓

jslib 플러그인 우회 및 매니저 중심 동기화

WebGL 네이티브 플러그인(.jslib)을 구축하여 브라우저 클립보드 복사 기능을 먼저 성공적으로 구현
추가로 브라우저의 fullscreenchange 이벤트 리스너를 연동하여 인게임과 전체 화면 상태를 동기화 성공

프레임 드랍

Optimization

웹 리소스 경합으로 인한 렌더링 성능 저하

브라우저 환경에서 메인 페이지의 DOM 애니메이션(밤 하늘 효과)과 WebGL 엔진이 CPU/GPU 자원을 동시에 사용하여 급격한 프레임 드랍(30FPS 이하) 발생

↓

JavaScript 기반 배경 애니메이션 제어

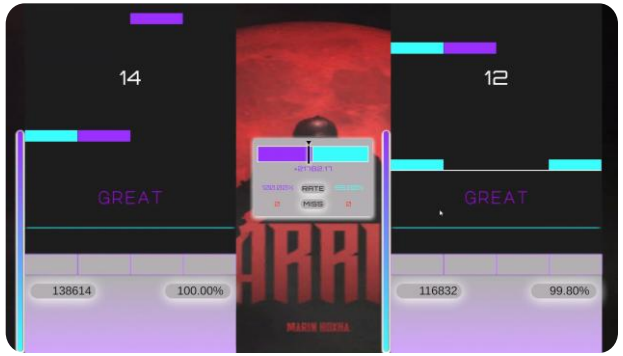
setBackgroundActive 함수로 마우스 진입 시 배경 효과를 정지 시켜고, WebGL에 자원을 집중하여 안정적으로 60FPS 확보
엔진의 Application.targetFrameRate 설정으로
프레임 제한 해제

Project 02

InMusic

Fusion2 기반 1:1 멀티플레이 리듬 게임

프로젝트 개요



기간: 6개월
인원: 2명

프로젝트 소개

Unity 6를 기반으로 구현한 Photon Fusion 2를 활용한 1:1 멀티플레이 리듬게임

Photon Fusion 2와 외부 API 및 백엔드 DB를 연동하여 클라이언트-서버 아키텍처의 기술적 도전

TECH STACK

Unity 6 | Photon Fusion 2 | Xampp | API | Database

주요 기능

Photon Fusion 2 기반 멀티플레이 환경 구축

Shared 모드 기반 P2P 통신 구조를 설계하고, RPC 메서드를 통해 실시간 이벤트 동기화 구현

유연한 UI/UX 및 데이터 관리 시스템 설계

무한 스크롤 방식의 음악 목록 UI 구현

외부 API 연동 및 백엔드 데이터베이스 통합

스팀 API를 연동하여 사용자 정보를 획득하고, Xampp(PHP, MySQL) 기반의 서버 환경을 구축하여 개인 기록 동기화

KEY ACHIEVEMENT

클라이언트-서버 아키텍처 확장을 기술적 도전

- 기존 Json 기반 로컬 관리 방식을 MySQL DB 기반으로 전환
- 동기화 과정에서의 UI 프리징 현상 해결 및 데이터 유실 방지
- 팀원 간의 기술적 의견 차이를 조율하여 리듬 게임 장르에 맞는 네트워크 변수 동기화 알고리즘 도입

상세

Notion Link

Photon Fusion 2 기반 멀티플레이 환경 구축

Photon Fusion 2 기반 멀티플레이

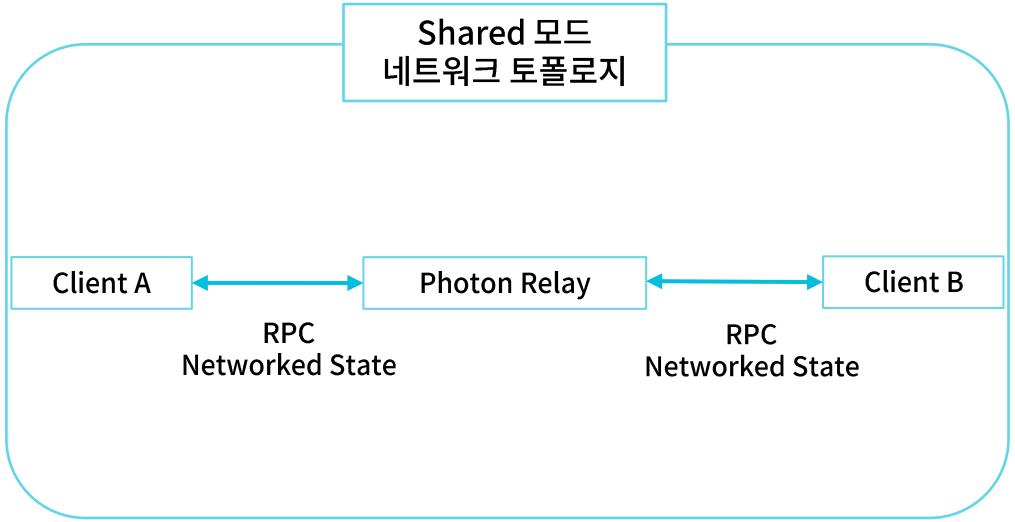
Photon 엔진의 Fusion 2 패키지를 활용하여 멀티플레이 환경 구현

Shared Mode 토폴로지 네트워크

별도 서버 구축 없이 Photon Cloud 기반 멀티플레이 환경 구현
Shared Mode를 활용하여 세션 및 객체 동기화 관리

상태 및 이벤트 기반

상태 데이터는 Networked 변수로 동기화
단발성 이벤트는 RPC를 통해 모든 클라이언트에 전파



유연한 UI/UX 및 데이터 관리 시스템 설계

사용자 친화적 데이터 설계

사용자가 외부 음악 파일(BMS, 음원, 미디어 리소스)를
로컬 폴더에 추가하면 게임 내에서 동적으로 인식 후
플레이 가능

슬라이딩 윈도우

화면에 노출되는 7개와 상하 영역 각 5개를 포함해
총 17개의 UI 오브젝트 고정 생성
데이터 슬라이딩 윈도우를 통해 기존 17개 객체 데이터
의 인덱스만 교체하여 반영

무한 스크롤 연출

스크롤 위치가 특정 임계점에 도달하는 순간 UI 전체
좌표를 반대편 끝에 배치
코루틴을 활용하여 시각적으로 끊김 없는 부드러운
스크롤 제공

외부 API 연동 및 백엔드 데이터베이스 통합

Steam SDK를 활용한 Steam API 연동

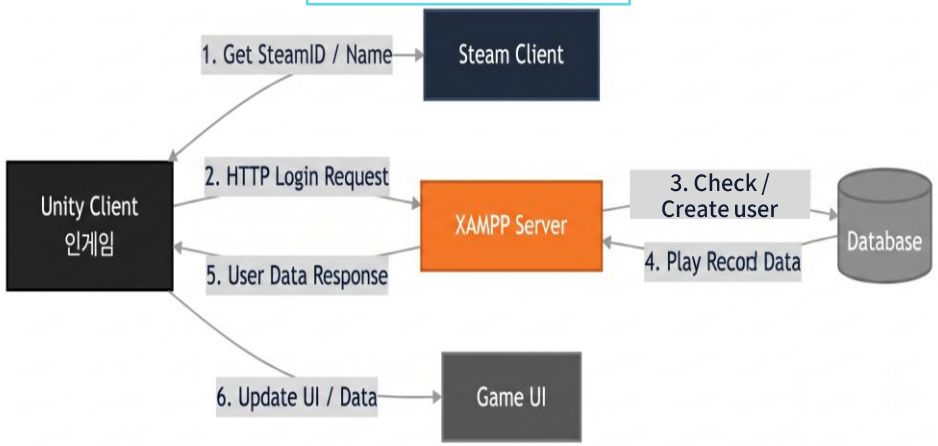
Steam SDK를 적용하여 사용자의 Steam 닉네임 및 Steam ID를 조회하고 게임 내 사용자 정보를 연동

Xampp 기반 웹 서버 및 DB 환경 구축

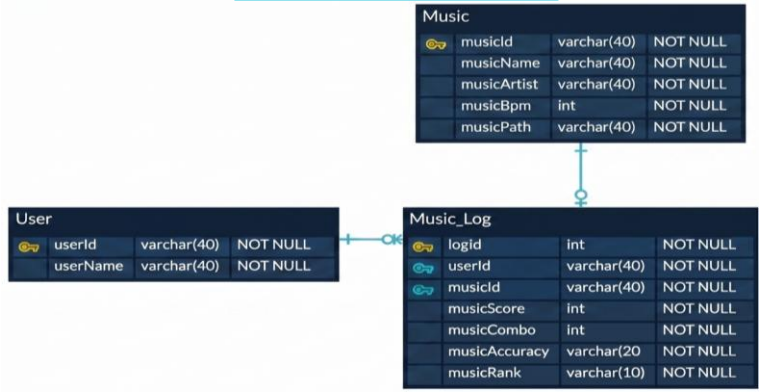
Apache 웹 서버와 MySQL DBMS를 활용하여 서버 및 데이터베이스 환경 구성

Steam ID와 음악 ID를 기본키로 하여 음악 기록 테이블 구축

흐름도



DB 테이블 구조



이슈 해결 – 비동기 로딩 및 네트워크 권한

비동기 로딩

Sync

비동기 프로세스 간의 실행 순서 불일치

백엔드 DB에서 가져오는 속도와 유니티 내부에서 리소스를 읽는 속도가 달라, 데이터가 도착하기 전에 화면이 출력되어 정보가 누락되는 문제 발생

↓

2단계 비동기 데이터 가공 프로세스 구축

타이밍을 확실히 맞추기 위해 로직을 두 단계로 분리

- 1단계: 로컬 음악 파일들을 읽고 음악 데이터 객체 준비
- 2단계: 서버 데이터 수신 후 데이터 객체에 할당 및 UI 반영

네트워크 권한 제어

Authority

네트워크 권한 체계로 인한 기능 구현 제약

Photon Fusion의 Shared 모드 특성상 모든 플레이어가 동등한 권한(Client)을 가짐. 방장은 다른 플레이어의 오브젝트를 조작할 수 없어 방장 권한 양도와 추방할 수 없는 한계가 존재

↓

RPC 브로트캐스팅 권한 우회 및 자가 제어

1:1 환경을 이용해 상대방이 보낸 신호를 자신이 처리하는 방식으로 우회

- 브로드캐스팅으로 신호 전달 후 본인 신호는 무시