

CASTLE BREAKER

조립식 오펜스 대전

GAME START

v0.1 오프라인 프로토타입 · 가상 플레이어 대전

개인 프로젝트

캐슬 브레이커 컨셉 기획서

설계하는 게임 / 조립식 1:1 라인 오펜스

문서 최초 작성	문서 최종 수정
20260806	20260807
박도원	박도원

목차

개요 – 2P

1. 목차
2. 로그
3. 문서 계층

현황 – 5P

1. 착수 시점 현황

시작 동기 – 6P

1. 문제의식과 해법

목표 – 7P

1. 프로젝트 목표

핵심 요소 – 8P

1. 핵심 요소 3가지

재미 요소 – 9P

1. 설계 × 읽기 매트릭스
2. 재미가 나는 순간

주요 루프 – 11P

1. 코어, 인게임, 아웃게임 루프

비주얼 – 12P

1. 비주얼 방향

범위 – 13P

1. 타깃, 플랫폼, 범위

제작 과정 – 14P

1. 마일스톤 5단계
2. 단계별 완료 기준

제작 관리 – 16P

1. 관리 방식

개요 – 로그

순번	작성일	작성자	작성 내용	비고
1	2026-08-06	박도원	Ver 0.1 작성	조립식 라인 오펜스 컨셉 초안
2	2026-08-06	박도원	기술/범위 확정	웹, TypeScript, 결정론 코어, v1 범위 결정
3	2026-08-07	박도원	컨셉 확정	기획서 v1.0 착수, 메인 기획 시작
4				
5				
6				

개요 – 문서 계층



컨셉 → 메인 → 세부 순으로 상위 문서의 결정을 목표로 받아 구체화함.
강조 표시 = 이 문서

현황 – 착수 시점 현황

구두 명세와 장르 경험만 있고 구현물은 없는 상태임.

가진 것

레퍼런스 장르

팔라독/냥코 대전쟁(라인 전투), 커스텀 로봇(조립)

AI 구현 체계

기획 1인이 명세를 쓰고 AI 에이전트가 구현하는 방식

웹 기술 경험

TypeScript, Vite, React, Canvas 2D

없는 것/제약

플레이 가능한 빌드

수치/규칙이 전부 구두 상태임

온라인 인프라

서버, 계정, 배포 환경 없음, 무료 플랜 수준 예산

아트 여력

그림/폰트 제작 인력이 없음

시작 동기 – 문제의식과 해법

기존 라인 전투의 빈칸

뽑기 중심

유닛을 뽑아 쓰는 구조라 설계 개입이 없음

정답 편성

스텟표를 외우면 반복 대전의 변수가 사라짐

이 프로젝트의 답

파츠 조립

유닛 카탈로그 대신 파츠로 병기를 직접 설계함

블라인드 배팅

상대 소환을 모른 채 예산을 걸어 읽기 싸움을 만듦

목표 – 프로젝트 목표

뽑는 게임이 아니라 설계하는 게임. 조립 결과가 승패의 절반을 정하는 8~14분 1:1 대전을 웹에서 완성함.

6종

매치에 들고 갈 병기 수

8~14분

1판 길이

M0~M4

계산기 → 온라인 5단계

의도 스텟과 가격이 같은 조립에서 파생되어 선택의 비용이 숫자로 읽히게 함.

근거 측정 기준은 M0~M4 수용 기준 통과 여부와 1판 시간(실시간 최대 600초 + 선택 시간)임.

핵심 요소 – 핵심 요소 3가지

조립 경제

가격도 스텟도 파츠 합계, 강하면 비싸거나 약점이 생김

의도 비용 대비 성능 자체가 설계 문제가 되게 함

근거 죽은 파츠가 없어야 선택이 성립함

블라인드 동시 배팅

선택 시간에 상대 소환을 모르고 예산을 걸음

의도 상성 순환 위에서 읽기 싸움을 만듦

근거 정답 편성이 없어야 반복 대전이 가능함

입력 없는 실시간

전투 구간에 입력이 없어 동기화가 자명함

의도 무료 호스팅에서도 지연 없는 온라인을 확보함

근거 서버 비용 제약

재미 요소 – 설계 × 읽기 매트릭스

상대 읽기 →

예측만 있음

고정 병기로 소환 타이밍만 읽음

목표 지점

내 설계가 상대 예측에 맞물림

재미 없음

정답 편성이 굳어 반복 대전이 단조로움

설계만 있음

상대 변수 없이 계산기 놀이에 그침

설계 자유도 →

재미 요소 – 재미가 나는 순간

조립 중

파츠를 붙일 때마다 가격과 스텟이 함께 움직임

공개 순간

양측 소환 목록이 뒤집혀 한 라운드 손익이 정해짐

상성 순환

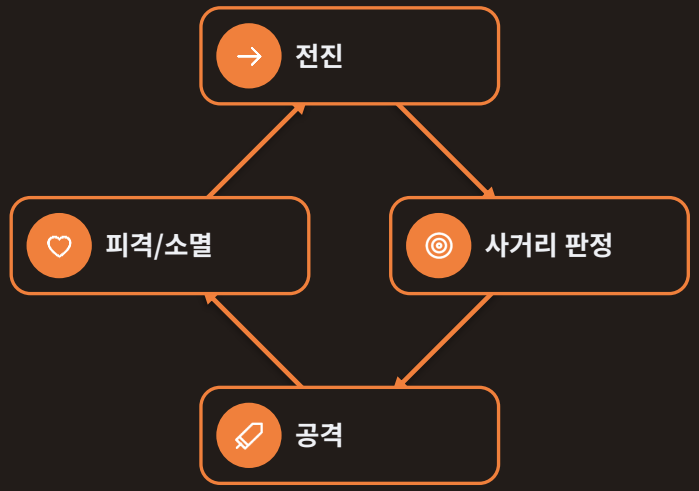
장거리 → 근접 물량 → 고기동이 서로 잡음

후반 판돈

텀이 길어져 한 번의 선택이 840원을 좌우함

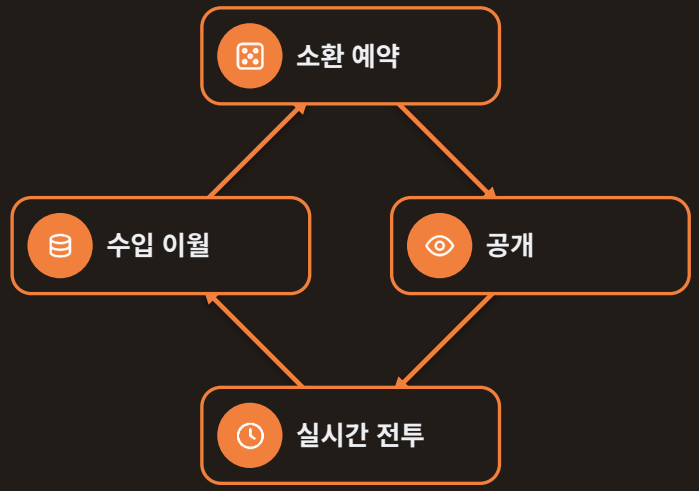
주요 루프 – 코어, 인게임, 아웃게임 루프

코어 루프 (초 단위)



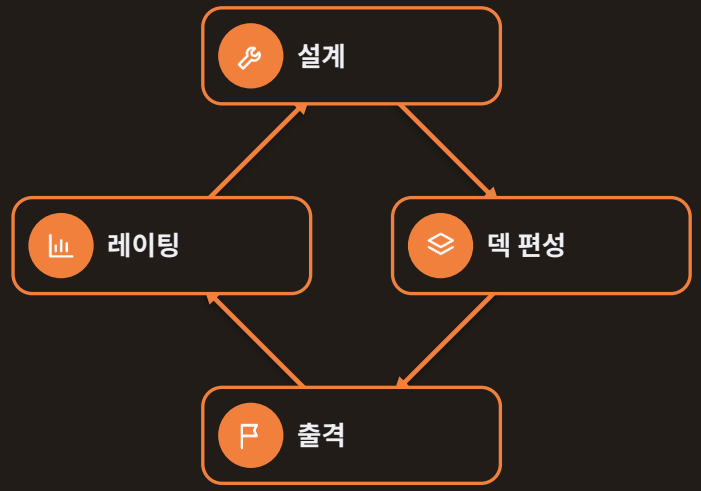
입력 없이 30Hz 틱으로 자동 진행함

인게임 루프 (라운드 단위)



R1~R7, 턴 30초에서 120초로 늘어남

아웃게임 루프 (매치 단위)



결과를 보고 병기를 다시 설계함

비주얼 – 비주얼 방향

그림 제작 여력이 없어 코드로 그리는 도형 아트로 정함.

톤과 화풍

미니멀 기하 플랫

아웃라인/그라디언트 없이 단색 면만 사용함

진영 보색

P1 청록, P2 주황으로 난전에서도 아군/적군이 구분됨

판독성 규약

기본 도형 조합

사각형, 삼각형, 원만 쓰고 외부 이미지 파일은 0개로 함

명도 위계

다리(어두움)에서 특수(밝음)까지 카테고리별로 고정함

범위 – 타깃, 플랫폼, 범위

항목	결정	이유
타깃	지정 소수 인원의 반복 대전	계정/보안 부담을 최소화하는 내부용 전제임
플랫폼	웹, 데스크톱 우선, 최소 1280×720	마우스 중심 조작, 모바일은 범위 밖임
기술	TypeScript, Vite, React 18, Zustand, Canvas 2D	전투는 Canvas, 비전투 화면은 React로 분리함
범위	AI 연습/온라인 1:1, 조립, 덱, Elo, 튜토리얼	리플레이, 관전, 모바일은 후순위임

제작 과정 – 마일스톤 5단계

1

M0 계산기

파츠 데이터와 조립 검증

2

M1 전투

결정론 코어와 로컬 전투

3

M1.5 검증

헤드리스 밸런스 시뮬레이터

4

M2~M3 화면

아트/조립 연출/셸 화면

5

M4 온라인

서버 권위 매칭과 저장

제작 과정 – 단계별 완료 기준

단계	완료 기준	검증 방법
M0	대표 빌드 스텟이 문서 값과 일치, 정수 연산만 사용	단위 테스트(검산 스크립트)
M1	같은 시드/명령이면 최종 해시가 동일	두 번 실행 비교
M1.5	BAL-1~3 승률 합격선 충족	고정 봇 100판 이상
M2~M3	메인→격납고→전투 순환, 외부 에셋 0개	수동 플레이
M4	두 브라우저로 완주, 전투 중 트래픽 0	2클라이언트 실측

제작 관리 – 관리 방식

항목	방식	이유
명세	단일 기획서(v1.0)와 상수 한 곳(constants)	AI 구현자가 모호함 없이 구현하기 위함
일정	마일스톤별 수용 기준 통과 시 다음 단계	단계마다 검산 가능한 합격선이 필요함
버전	git 저장소, 규칙 개정은 결정 기록(§15)에 남김	원안과 달라진 이유를 추적하기 위함
검수	check/sim 스크립트와 죽은 파츠 검사	사람 눈이 아닌 수치로 밸런스를 확인함
도구	Claude(구현), Vitest, Vite, 헤드리스 시뮬레이터	기획 1인 체제에서 구현/검증을 자동화함

제작 관리 – 리스크와 대응

⚠ 곱연산 폭주나 열등 파츠로 선택지가 무의미해질 수 있음	→ 🛡 죽은 파츠 검사와 교환비 측정을 검수 항목으로 둠
⚠ 무료 플랜에서 실시간 서버를 둘 수 없음	→ 🛡 결정론 시뮬로 라운드 경계 교환만 하는 구조를 택함
⚠ 플랫 스타일에서 밀집 유닛이 한 덩어리로 보일 수 있음	→ 🛡 네거티브 스페이스와 명도 위계를 판독성 규약으로 둠

작업 계획 – 메인 기획 착수 전 작업

1

기획서 v1.0

수치/상태 전이/스키마 문서화

2

파츠/경제 초안

파츠 18종과 가격 곡선 정리

3

동기화 방식

결정론 락스텝과 서버 구성 결정

4

마일스톤

M0~M4 수용 기준 설정

작업 계획 – 완수 방안

1

레퍼런스 분해

라인 전투와 조립 게임의 재미 요소를 분리해 정리할 예정임

플레이 메모

의도 두 장르의 장점만 가져오려 함

2

명세 문서화

AI가 추가 질문 없이 구현할 수준으로 쓸 예정임

오펜스게임_기획서.md

근거 구현자가 AI 에이전트라 모호함이 곧 재작업임

3

수용 기준

단계마다 검산 가능한 합격 기준을 정할 예정임

npm run check / sim

4

계산기부터

M0 조립 계산기에서 시작해 M4 온라인까지 쌓을 예정임

AI 에이전트