바운스페이스 — 세부기획서: UI/UX와 아트/사운드

> 재구성 기획서 (역설계) / 작성 기준 시점: 2026-11-06 (프로젝트 시작 시점) / 문서 버전: Ver 0.1 프로젝트 ID: P005 / 근거 수준: A / 작성: Claude (재구성) 이 문서는 결과물과 제작 기록으로부터 기획 의도를 역추정해 당시 작성한 것처럼 재구성한 것이다. 확인된 사실과 추정을 [근거]/[추정] 태그로 구분한다.

1. 목적과 범위

목적

조작, 화면, 캐릭터 시점 표현, 리소스 예산, 이펙트, 사운드를 구현 가능한 수준으로 정한다.

범위

입력, HUD, 로비 화면, 캐릭터와 1인칭 표현, 베벨 도형 리소스, 이펙트, 합성 사운드, 폴리싱(P1~P6). 규칙은 02, 네트워크는 03 네트워크 문서.

2. 설계

3.1 조작과 입력 처리

- 의도(Why): 한글 입력 환경에서도 이동이 되고, 채팅과 조작이 충돌하지 않게 한다.
- 설계(How): 마우스 시점, `W A S D` 이동, `Space` 점프, 좌클릭 충격파, 우클릭 특수 능력, `E` C4 회수, `Q` 부활 대기 중 특수 능력 변경, `Enter` 전체 채팅, `T` 팀 채팅, `Tab` 점수판, `Esc` 메뉴. 이동 키는 `event.key`가 아니라 `event.code`로 읽는다. 채팅은 실제 `<input>`을 쓰고 열면 게임 입력을 막으며 `event.isComposing`을 검사한다. 마우스 잠금을 거는 클릭과 잠금 직후 0.25초는 발사하지 않는다(이후 요청 반영). 채팅 중에도 마우스 잠금 유지(이후 요청).
- 필요(What): 포인터 잠금, `ui/chat.ts`, 입력 모듈.
- 작업 방법: 한글 입력 수동 체크리스트(WASD 이동, 조합 중 Enter 전송, 채팅 중 정지, 닫은 뒤 클릭 한 번으로 복귀).
- 완료 기준: 체크리스트 통과. 실제 한글 IME는 자동화 불가라 사용자 확인 필요로 기록됨.
- 리스크/대안: 마우스 잠금 직후 오발 → 0.25초 발사 억제.
- 근거: [근거: `IMPLEMENTATION_PLAN.md` 2.8, 2.9, 7장, `HANDOFF.md` T-019, 로그 11-08]

3.2 HUD와 화면 구성

- 의도(Why): 사용 횟수가 이 게임의 핵심 자원이므로 한눈에 보여야 한다. 정보는 일반 경쟁 FPS의 배치를 따른다.
- 설계(How): 조준선, 지상/공중 사용 횟수(오른쪽 아래에 두 탄창 두 줄), 상태 아이콘, 킬 로그(오른쪽 위, 킬러/사인 아이콘/대상, 내 킬 강조), 점수판, 거점 게이지, 안전 구역 표시, 목표 킬(데스매치 계열만), 부활 대기 카드, 결과 화면과 MVP, 머리 위 이름표(아군은 항상, 적은 시야 안 45m 이내). 화면: 시작, 방 목록, 대기실(팀별 칸, 방장의 팀 옮기기), 결과. 한국어/영어(`ui/i18n.ts`, 번역 누락 테스트). 궤도 예측선은

계획서 6.4에서 유지 여부를 보기로 했으나 사용자 결정으로 제외되었다. 특수 능력 선택 원판(`ui/loadout-picker.ts`)은 경기 시작과 부활 대기 중 Q로 열리고 떠 있는 동안 다른 조작을 받지 않는다(서버 규칙 `loadoutLocked`).

- 필요(What): SVG 아이콘(사인 5종/특수 능력 5종), 번역표, 음소거/음량 막대(왼쪽 위 `.corner-controls`).
- 작업 방법: 기능 구현 → P5 정보 배치 → P6 전면 재디자인(달 탐사선 계기판: 가는 선, 눈금, 잘린 모서리, 글꼴 Oxanium/Pretendard, 이후 Barlow Condensed) → 사용자 요청 반영.
- 완료 기준: 현재 몇 회를 더 쓸 수 있는지 한눈에 보이고 언어 전환이 동작한다.
- 리스크/대안: 계획서 6.4 "한글 웹폰트 없음"과 P6의 Pretendard 도입이 상충한다(P6는 필요한 글자 조각만 내려받는다고 기록). 최종 용량 영향은 미확인.
- 근거: [근거: `IMPLEMENTATION_PLAN.md` 6.4, `POLISH_PLAN.md` P5/P6, `HANDOFF.md` T-037, 로그 11-07~08]

3.3 캐릭터와 시점 표현

- 의도(Why): 다른 플레이어가 어디를 보는지 몸에서 읽혀야 조준 예측과 협동이 된다.
- 설계(How): 시점 추종 비율 — 머리는 yaw 100%/pitch 50%, 상체 yaw 100%/pitch 30%, 팔/무기 yaw 100%/pitch 나머지 20%를 더해 총 100%, 하체는 시점과 60도 이상 벌어지면 따라 돈다(pitch 없음). 노드 구조 `root > hips > (legL>shinL, legR>shinR, torso>(head, armL>forearmL, armR>forearmR>weaponSocket))`, 스키닝 없이 부위 노드를 직접 돌린다. 코드 보간 동작 9종: `idle`, `run`, `jump`, `air`, `land`, `knocked`, `spin`, `boundUpper`, `boundLower`. 구현은 부위별 `InstancedMesh`. 개인전은 흰 우주복(`#F2F4F7`)과 이름표만, 팀전은 팀 색과 보조 표식(색각 차이 대비). 팀전에서 1인칭 손/팔도 팀 색.
- 필요(What): `render/shapes.ts` 키트, 팀 색 재질.
- 작업 방법: 계층 구성 → 시점 추종 → 애니메이션 → 팀 색.
- 완료 기준: 캐릭터 1명 삼각형 3,000개 이하, 계획서 2.10 규칙대로 보인다.
- 리스크/대안: 코드 애니메이션이라 어색할 수 있음 → 사람 확인 필요.
- 근거: [근거: `IMPLEMENTATION_PLAN.md` 2.7, 2.10, 6.2, `HANDOFF.md` T-033, 로그 11-07~08]

3.4 1인칭 무기 표현과 피격 연출

- 의도(Why): 충격파를 맞히는 느낌이 가장 중요한 쾌감이므로 발사/적중 연출을 강화하되 네트워크 비용을 늘리지 않는다.
- 설계(How): 오른손 충격파 건, 왼손 특수 장비(1인칭에서만, 다른 사람에게는 왼손 장비를 보이지 않음 — 사용자 결정). 발사 문법: 코어 맥동 → 총구 섬광 → 빛줄기 → 반동/복귀 → 착탄 고리, 불꽃, 탄흔. 피격 시 시야각 확대, 빠른 비행 시 속도선, 피격 방향 표시, 적중 표시. 로켓 피격 시 화면 완전 회전(D10), 회전 완화 시 20%. 연출은 서버가 이미 보내는 위치, 속도, 상태, 사건에서 클라이언트가 만든다. 파티클 위치 동기화, 킬캠, 핑 표시는 제외.
- 필요(What): `render/view-model.ts`, `render/earth-fx.ts`, 파티클, 화면 흔들림.
- 작업 방법: P2(양손/발사) → P3(피격) → P4(발자국, 먼지, 호흡음, 탄흔).
- 완료 기준: 사용자가 충격파 적중을 분명히 느낀다. 손맛은 사람 확인 필요로 기록됨.

- 리스크/대안: 멀미 → 회전 완화 설정과 시작 화면 안내.
- 근거: [근거: POLISH_PLAN.md 1~3장, IMPLEMENTATION_PLAN.md 2.3, HANDOFF.md 로그 11-07]

3.5 리소스 예산과 제작 규격

- 의도(Why): 내장 그래픽에서도 60FPS, 첫 다운로드 5MB 이하를 지킨다.
- 설계(How): 모든 모델을 베벨 도형 키트로 조립. 키트: bevelBox(RoundedBoxGeometry), bevelCylinder(LatheGeometry), capsule, sphere, torus. 베벨 반지름 작은 부품 0.02m, 몸체 0.04m, 구조물 0.15m. 같은 치수는 캐시 공유, 정적 물체는 병합 또는 InstancedMesh. 예산:

항목	상한
캐릭터 1명 / 1인칭 팔/무기 / 장비 1개 / 소품 1개 / 지형 전체	삼각형 3,000 / 2,000 / 800 / 300 / 40,000
텍스처	512px 이하, 전체 8장, 합계 1MB 이하, .webp
드로우 콜 / 화면 삼각형	150 / 150,000
그림자	태양광 그림자 맵 1장 1024px, 후처리 없음
첫 다운로드	5MB 이하(사운드 포함)

외부 모델은 첫 버전에 불필요하며 나중용 규격으로 .glb, 1유닛=1m, Y축 위, 정면 -Z를 둔다. 하늘의 지구 텍스처는 코드로 생성했다. 외부 에셋을 넣으면 assets/CREDITS.md에 기록한다. - 필요(What): 리소스 목록 — 우주복 캐릭터 1, 1인칭 팔 1, 충격파 건 1, 특수 장비 5, 투사체 4, 지형 맵 1, 기지 모듈 2, 소품 5~8, 하늘 1. - 작업 방법: 장면 → 지형 → 구조물 → 소품 → 하늘, 매 단계 드로우 콜 측정. - 완료 기준: 22인 전투 시점에서 위 예산 준수. - 리스크/대안: 도형이 밋밋함 → 정점 색, 그림자, 이펙트. 실제 22인 렌더 실측값은 기록 없음. - 근거: [근거: IMPLEMENTATION_PLAN.md 6.1~6.6, HANDOFF.md T-033~T-035, 로그 11-06]

3.6 사운드와 음악

- 의도(Why): 달에는 공기가 없으므로 헬멧 안의 진동으로 듣는 소리를 만들고, 파일 없이 합성해 용량/라이선스를 없앤다.
- 설계(How): 전부 Web Audio 합성(audio/sound.ts), 위치 기반 재생, 음량 설정. 목록: 기본 3~5, 이동 4~6, 무기 8~10, 상태 4, UI 6~8, 배경 2. 배경 음악은 audio/music.ts로 별도 구현하고 audio/instruments.ts로 악기를 합성한다. 최종 형태는 사용자 지시에 따라 낮은 음역의 빠른 저음 여러 겹(126BPM, 마림바 선율 G2~C4와 한 옥타브 아래 겹침, 16분음표 저음, 낮은 탐탐, 저역 통과 1kHz)에서 다시 4도 내려 G장조/650Hz로 조정되었다. 큰 소리가 나면 음악이 잠깐 줄어든다. 효과음은 실로폰 계열 변경 뒤 원래대로 되돌렸다.
- 필요(What): 이벤트별 소리 목록, 음량, 음악, 음소거 UI(저장됨).
- 작업 방법: 합성 → 사용자 청취 → 요청 반영. 자동 검증 불가.
- 완료 기준: 모든 필수 이벤트에 소리가 있고 첫 다운로드가 5MB 이하.
- 리스크/대안: 취향 차이로 음악을 5회 이상 교체했다(11-08 로그). 거슬리는 지속음(무작위 패드 음, 체공 중 호흡 잡음)은 제거되었다.
- 근거: [근거: IMPLEMENTATION_PLAN.md 6.5, HANDOFF.md T-038, 로그 11-07~08]

3. 데이터/리소스 명세

항목	필드/규격	예시
i18n 문구	key, ko, en	번역 누락 테스트로 검증
팀 색	02 데이터 문서 참조	`#E5484D` 등
사운드 파일 규격 (교체용)	WAV 48kHz 16bit 원본, `.ogg` 배포, 효과음 -14 LUFS/배경 -22 LUFS	`sfx_shock_fire_01.ogg`
아이콘	SVG 24x24, `icon_{이름}.svg`	사인 5종, 특수 능력 5종
UI 재질	Codex 생성 키아트, 금속, 먼지, 빛 이미지 → `src/material.css` 마스킹/합성	11-08 변경

[근거: `IMPLEMENTATION_PLAN.md` 6.4, 6.5, `HANDOFF.md` 로그 11-08]

4. 작업 순서

1. 도형 키트와 장면(T-005) → 2. 1인칭 조작(T-006) → 3. 캐릭터, 무기, 맵(T-033~T-035) → 4. 이펙트, HUD, 사운드(T-036~T-038) → 5. 폴리싱 P1~P6 → 6. 사용자 요청별 개선. [근거: `HANDOFF.md`, `POLISH_PLAN.md`]

5. 검수 기준

- 드로우 콜, 삼각형, 첫 다운로드 용량 측정. 사람 확인 필요 항목: 사운드 청취, 아트 품질, 손맛, 한글 IME. [근거: `HANDOFF.md` 로그 11-06~07]
- 래스터 이미지를 쓰지 않은 P6 단계와 Codex 이미지를 쓴 11-08 단계가 섞여 있어 최종 UI 구성은 코드로 재확인이 필요하다. 미확인.

바운스페이스 — 세부기획서: 네트워크와 데이터

> 재구성 기획서 (역설계) / 작성 기준 시점: 2026-11-06 (프로젝트 시작 시점) / 문서 버전: Ver 0.1 프로젝트 ID: P005 / 근거 수준: A / 작성: Claude (재구성) 이 문서는 결과물과 제작 기록으로부터 기획 의도를 역추정해 당시 작성한 것처럼 재구성한 것이다. 확인된 사실과 추정을 [근거]/[추정] 태그로 구분한다.

1. 목적과 범위

목적

프로토콜, 스냅샷 필드, 세션/입력 방어, 수치 데이터, 배포/운영을 구현 가능한 명세로 정한다.

범위

메시지 종류와 인코딩, 대역폭, 접속/재접속, 방어 규칙, `balance.ts` 수치 표, 서버 공개 방법. 렌더 연출은 제외.

2. 설계

3.1 프로토콜 구성

- 의도(Why): 대기실 메시지는 읽기 쉬운 JSON, 경기 중 고빈도 메시지는 바이너리로 대역폭을 아낀다.
- 설계(How): 단일 WebSocket으로 로비와 경기를 모두 처리한다. C→S JSON: `hello`, `listRooms`, `createRoom`, `joinRoom`, `leaveRoom`, `updateSettings`, `setTeam`, `setLoadout`, `startGame`, `chat`, `ping`. S→C JSON: `welcome`, `pong`, `roomList`, `roomState`, `chat`, `error`, `matchStart`, `matchEnd`, `events`. 바이너리: `input`(30Hz), `snapshot`(20Hz). 입력 패킷은 메시지 종류 1B, 순번 2B, 이동/버튼 비트 1B, yaw/pitch 각 2B(발사 시 조준 방향 6B 추가, 기록상 조준 포함 14B). 프로토콜 버전은 현재 4(`PROTOCOL_VERSION`).
- 필요(What): `protocol/messages.ts`, `binary.ts`, `events.ts`, 왕복 인코딩 테스트.
- 작업 방법: 인코더 → 디코더 → 왕복 테스트 → 양자화 오차 범위 테스트.
- 완료 기준: 왕복 테스트와 양자화 오차 테스트 통과.
- 리스크/대안: 패킷 손실 → 전송 계층을 인터페이스로 분리해 WebTransport 추가 여지. 사건은 처음 JSON이었다가 이진(`events.ts`)으로 바뀌었다(프로토콜 3).
- 근거: [근거: `IMPLEMENTATION_PLAN.md` 4.3, 4.4, `HANDOFF.md` T-010, 로그 11-08, `messages.ts`]

3.2 스냅샷 데이터

- 의도(Why): 22명 방에서 스냅샷을 약 500B로 유지해 대역폭을 낮춘다.
- 설계(How): 머리글 8B(종류 1, 틱 4, 처리한 입력 순번 2, 플레이어 수 1). 플레이어당 22B 고정.

필드	형식	크기
플레이어 번호	u8	1B
위치 x,y,z	int16, 1/128m (±256m)	6B
속도 x,y,z	int16, 1/256m/s (±128m/s)	6B
yaw, pitch	u16 / int8	3B
상태, 보호, 횟수, 묶임, 방패, 제트팩, C4	u16	2B
몸 회전	int8 x4	4B

22명 기준 492B, 클라이언트당 수신 약 9.8KB/s, 서버 송신 약 216KB/s(계획서 추정 1.7Mbps). 본문을 한 번만 만들고 클라이언트마다 입력 순번 위치만 바꿔 보낸다. 투사체는 스냅샷에 넣지 않고 사건으로 전달한다. 가변 길이 스냅샷은 시도 후 버그 위험으로 되돌려 22B 고정으로 복구했다(프로토콜 4). - 필요(What): 양자화 함수, 입력 확인 번호, 사건 인코더. - 작업 방법: 22명 스냅샷 크기를 테스트로 고정(실측 404B 기록, 투사체 제외/계획서 초안 대비 차이는 미확인). - 완료 기준: 실측 송신량 기록: 봇 22명 서버 3분, 약 3Mbps(스냅샷 1.75, 사건 0.9, 상태 0.35). - 리스크/대안: 양자화 범위(±256m, ±128m/s)를 넘는 값은 maxSpeed 40으로 제한되어 문제 없음. - 근거: [근거: `IMPLEMENTATION_PLAN.md` 4.4, `HANDOFF.md` 로그 T-010, 11-08, `balance.ts`]

3.3 접속, 세션, 재접속

- 의도(Why): 계정 없이도 새로고침/끊김에서 자리를 되찾고, 한 사람이 둘로 늘어나지 않게 한다.

- 설계(How): 첫 접속에 서버 입장 코드(ACCESS_CODE)와 닉네임을 입력한다. 브라우저 localStorage의 임의 playerToken으로 세션을 식별하고 토큰당 소켓 1개/방 1개만 허용한다. 끊겨도 30초 자리 유지. 같은 브라우저 새 탭은 이전 탭을 대체하고 이전 탭은 안내 화면으로 바뀐다. 시크릿 창/다른 기기는 별개 사람(D4).
- 필요(What): 서버 세션 맵, .env.host의 코드.
- 작업 방법: 서버 테스트로 중복/재접속 검증, 브라우저로 탭 대체 확인.
- 완료 기준: 새 탭이 이전 탭을 대체, 30초 안 재접속 시 같은 자리, 번호, 킬 유지.
- 리스크/대안: 입장 코드가 localStorage에 평문 저장된다. 코드 시도 횟수는 T-040에서 제한.
- 근거: [근거: IMPLEMENTATION_PLAN.md 2.6, HANDOFF.md T-016, 로그 11-06]

3.4 입력 방어

- 의도(Why): 링크가 새어 나가도 서버가 죽거나 속지 않게 한다.
- 설계(How): 서버 권위 판정, 메시지 크기/빈도 제한, 입력값 범위 검사, 조준 방향은 단위 벡터 확인, 채팅은 200자/10초 5개 제한과 textContent 표시, 방 상한 8개, 연결 빈도/입장 코드 시도/입장 대기 제한.
- 필요(What): 잘못된 값 테스트 세트.
- 작업 방법: 잘못된 패킷 전송 테스트.
- 완료 기준: 잘못된 패킷에도 서버가 죽지 않는다.
- 리스크/대안: 방어 수준은 친구용 비공개 서버를 전제로 최소 수준.
- 근거: [근거: IMPLEMENTATION_PLAN.md 8장, HANDOFF.md T-040]

3.5 성능/부하 기준

- 의도(Why): 22인 방을 PC 1대로 감당한다.
- 설계(How): 22명 경기에서 30Hz 틱 처리 시간 p99가 8ms 이하. 봇 22명 10분 부하 테스트(tools/bots)로 측정.
- 필요(What): 봇 도구, 틱 시간 측정.
- 작업 방법: 서버를 별도 프로세스로 돌려 10분 측정.
- 완료 기준(기록된 결과): 틱 1800회/분 유지, 힙 12~25MB로 증가 없음. p99는 10구간 중 7구간이 8ms 이하, 3구간은 10.5~34ms(최대 163ms)로 조건부 완료. 2026-11-08 재측정에서 서버 3분 p99 4.3~4.9ms.
- 리스크/대안: 봇과 서버가 같은 PC라 실서버 재측정이 필요하다고 기록되었으나 그 뒤 기록 없음.
- 근거: [근거: HANDOFF.md 로그 T-039, 11-08]

3.6 서버 공개와 운영

- 의도(Why): 비용/계정 없이 친구가 링크로 접속하게 한다.
- 설계(How): 계획서 초안은 한국/일본 리전 VPS 1대, Node 프로세스 1개, systemd, Caddy HTTPS, ACCESS_CODE, DB 없음이었다. 실제 공개는 이 PC + Cloudflare Quick Tunnel(D21)로 정했고 npm run host(tools/host.mjs)가 서버와 터널을 함께 띄운다. 입장 코드는 .env.host(저장소 제외)에 최초 한 번 생성된다. 2026-11-10에 클로드 없이 켜고 끌 수 있는 bounspace-server.bat을 추가했다: 켜기(빌드

후 `host.mjs` 숨김 창 실행), 끄기(프로세스 종료와 포트 8788 정리), 상태 보기. 켜면 주소와 입장 코드 안내 블록이 뜬다. 터널 주소는 재시작마다 바뀐다.

- 필요(What): `cloudflared`(공식 배포본, 서명 확인), Node.js 24.16.0, 포트 8788.
- 작업 방법: `npm run build` 후 `npm run host`. `npm run host`는 빌드하지 않으므로 코드 변경 후에는 빌드가 필요하다(런처가 빌드를 포함).
- 완료 기준: 공개 주소로 WebSocket이 동작하고 브라우저 2개로 한 판 완주(이 PC 기준 왕복 약 8ms).
- 리스크/대안: 다른 네트워크의 실제 지연은 미측정. 필요 시 Oracle 무료 서버 이전. 런처의 1번 메뉴가 처음 반응하지 않은 문제는 한글이 든 배치 파일에 `chcp 65001`을 쓴 것이 원인으로 추정되어 화면 문구를 영어로 바꿔 해결했다(직접 재현은 하지 못했다고 기록).
- 근거: [근거: `HANDOFF.md` 로그 11-06, 세션 `claude_d14a8ff8`, `IMPLEMENTATION_PLAN.md` 8장]

3. 데이터/리소스 명세 (`balance.ts` 확정값)

분류	이름	값
틱	tickRate / snapshotRate	30 / 20
이동	gravity / runSpeed / groundAccel	1.5 / 3.375 / 16
이동	jumpSpeed / jumpAngleDeg / jumpLaunchSpeed / jumpSettleSeconds / maxSpeed	4.5 / 62.5 / 3.6 / 0.5 / 40
몸	playerRadius / playerHeight / eyeHeight / bolaUpperSplit	0.4 / 1.8 / 1.6 / 0.9
반사	restitution / jumpWallRestitution / landingSpeed	0.85 / 0.7 / 1.5
반사판	bumperKick / bumperTangentKeep / bumperThickness	13 / 0.4 / 0.6
충돌	playerCollideMinSpeed / playerCollideRestitution	3 / 0.85
횟수	groundUses / airUses / minUseInterval	1 / 3 / 0.35
충격파	shockSpeed / shockRadius / shockWorldRadius / shockRange	70 / 0.9 / 0.2 / 90
충격파	maxFireLead / shockImpulse / shockBlastRadius / shockBlastImpulseEdge / groundHitMinLift	0.1 / 16 / 2.7 / 6 / 0.125
로켓	rocketSpeed / rocketRadius / rocketRange / rocketHitImpulse	30 / 0.35 / 120 / 3
로켓	rocketSpin / rocketMaxSpin / rocketThrust / rocketDuration / rocketMaxAttached	0.3 / 8 / 12 / 2.5 / 3
볼라	bolaSpeed / bolaRadius / bolaRange / bolaDuration	45 / 0.4 / 60 / 3
제트팩 / 방패	jetpackMinSpeed / shieldDuration / shieldArc	8 / 0.8 / 120
C4	c4ThrowSpeed / c4Radius / c4ImpulseCenter / c4ImpulseEdge / c4RetrieveRange	18 / 10 / 22 / 8 / 2.5
판정	killCreditWindow / respawnDelay / spawnProtection	10 / 3 / 2
점령	neutralCaptureTime / mainCaptureTime / captureBonusPerExtra / captureMaxRate	15 / 45 / 0.5 / 3
구역	shrinkStart / shrinkEnd / shrinkMinRadius / outsideZoneLimit	120 / 240 / 12 / 3

[근거: `balance.ts`]

4. 작업 순서

1. 프로토콜과 왕복 테스트 → 2. 서버 틱 루프 → 3. 예측, 보정, 보간 → 4. 지연 시뮬레이션과 명중률 측정 → 5. 세션/방 → 6. 입력 방어 → 7. 22명 부하 → 8. 배포 빌드와 공개. [근거: `HANDOFF.md` T-010~T-042]

5. 검수 기준

- `npm run check`, `npm run e2e`, `npm run net:check -- 100 20 30 8`, `npm run bots -- --count 22` 통과. [근거: `HANDOFF.md` 5장]
- 계획서 3장과 `balance.ts` 수치 일치(예외: `HANDOFF.md` 3장 D20은 낡은 4.5로 표기). [근거: 00 문서 6절]

바운스페이스 — 세부기획서: 모드와 맵

> 재구성 기획서 (역설계) / 작성 기준 시점: 2026-11-06 (프로젝트 시작 시점) / 문서 버전: Ver 0.1 프로젝트 ID: P005 / 근거 수준: A / 작성: Claude (재구성) 이 문서는 결과물과 제작 기록으로부터 기획 의도를 역추정해 당시 작성한 것처럼 재구성한 것이다. 확인된 사실과 추정을 [근거]/[추정] 태그로 구분한다.

1. 목적과 범위

목적

같은 이동/넉백 규칙 위에서 모드 6종의 승패 조건과 맵 구조를 구현 가능한 수준으로 정한다.

범위

모드 6종, 점령전 거점, 태그매치, 안전 구역, 방 설정과 흐름, 맵 JSON, 봇. 이동/무기 규칙은 02 문서.

2. 설계

3.1 모드 일람과 시간 종료 판정

- 의도(Why): 모드마다 압박의 종류를 다르게 해 같은 조작으로 다른 판단을 요구한다. 어떤 모드든 제한 시간이 끝나면 결과가 나와야 한다(멈춤 방지).
- 설계(How):

모드	팀	부활	승리 조건	시간 종료 판정
서바이벌	0	불가	최후의 1인	살아남은 쪽
데스매치	0	가능	제한 시간 종료 시 킬 최다 또는 목표 킬	최고 점수, 동점은 공동 우승
팀 서바이벌	2~5	불가	마지막으로 인원이 남은 팀	살아남은 쪽
팀 데스매치	2~5	가능	팀 킬 최다 또는 목표 킬	최고 점수
팀 점령전	2~5	가능	마지막까지 주요 거점을 지킨 팀	중립 거점을 많이 가진 팀

모드	팀	부활	승리 조건	시간 종료 판정
팀 태그매치	2~5	공통 목숨	마지막까지 목숨이 남은 팀	목숨이 많은 팀

- 필요(What): `createRules` 인터페이스, 결과 화면, 점수판(Tab).
- 작업 방법: 모드별 규칙 구현 → 봇 시뮬레이션으로 전 모드 종료 테스트. 담당 AI, 규칙 해석이 결과를 바꾸면 사용자 질문.
- 완료 기준: 모든 모드가 승자 또는 무승부로 끝나고 멈춤 상태가 없다.
- 리스크/대안: 개인전 모드는 팀 0, 팀 모드는 2~5만 허용(1팀은 상대가 없어 제외).
- 근거: [근거: `IMPLEMENTATION_PLAN.md` 1.2, 2.5, `HANDOFF.md` T-027, T-032]

3.2 서바이벌과 안전 구역

- 의도(Why): 부활이 없는 모드는 서로 피해 다니면 끝나지 않으므로 교착 해소 장치가 필요하다(D11).
- 설계(How): 경기 시작 120초(`shrinkStart`)부터 240초(`shrinkEnd`)까지 안전 구역 반지름이 맵 전체에서 12m(`shrinkMinRadius`)로 줄어든다. 구역 밖에 3초(`outsideZoneLimit`) 연속이면 장외 탈락. 마지막 생존자들이 같은 틱에 탈락하면 무승부. 시간과 구역은 클라이언트가 계산해 전송량을 줄인다.
- 필요(What): `zoneRadiusAt(startRadius, elapsedTicks)`, 구역 벽 셰이더.
- 작업 방법: 순수 함수로 구현해 단위 테스트, 구역 표시는 클라이언트.
- 완료 기준: 부활 불가, 구역 축소와 표시, 3초 밖 탈락, 동시 탈락 무승부.
- 리스크/대안: 축소가 너무 빨라 불합리 → `balance.ts` 수치 조정.
- 근거: [근거: `IMPLEMENTATION_PLAN.md` 2.5, `balance.ts`, `modes/rules.ts` `zoneRadiusAt`, `POLISH_PLAN.md` 2장]

3.3 팀 점령전

- 의도(Why): 이동 중심 게임에서 "서 있어야 얻는 목표"를 만들어 위치 싸움에 방향을 준다. 공중에서는 점령되지 않으므로 착지와 이륙의 판단이 걸린다.
- 설계(How): 거점은 팀별 주요 거점 N개와 중앙 중립 3개. 범위 안에서 `GROUNDED`인 플레이어만 기여. 거점마다 `progress[team]`이 있고 값은 오르기만 한다(D5). 여러 팀이 동시에 있으면 각 팀 게이지가 동시에 오른다. 소유 팀은 자기 거점 게이지를 올릴 수 없다. 100% 도달 팀이 소유권을 얻고 그 거점의 모든 게이지가 0이 된다. 같은 틱 동시 도달은 범위 안 인원이 많은 팀, 그다음 낮은 팀 번호가 우선. 속도: 중립 15초/주요 45초(1인 기준), 추가 인원당 +0.5배, 최대 3배. 주요 거점을 잃은 팀은 탈락(D6)하고 전원 관전, 그 팀의 중립 거점은 중립으로 복귀, 빼앗긴 거점은 비활성. 부활 위치는 자기 주요 거점 또는 소유한 중립 거점 중 선택.
- 필요(What): 거점 배치 함수(원 위에 360/팀 수 간격), 중립 3개 고정 좌표, 거점 UI 원과 게이지.
- 작업 방법: 2~5팀 배치 생성 테스트 → 게이지 단조 증가 테스트 → 3팀에서 한 팀 탈락 후 계속 진행 테스트.
- 완료 기준: 2~5팀 배치 모두 생성, 공중 점령 불가, 게이지는 소유권 이전 외 감소 없음, 3팀 탈락 후 속행.
- 리스크/대안: 팀 수에 따른 맵 불균형 → 원형 대칭 배치.
- 근거: [근거: `IMPLEMENTATION_PLAN.md` 2.5, 4.5, 3장, `HANDOFF.md` T-030]

3.4 팀 태그매치

- 의도(Why): 인원이 다른 팀도 공정하게 붙게 하고, 대기자는 관전하며 참여감을 유지한다(D7).
- 설계(How): 필드에는 팀별 1명만 나온다. 모든 팀의 목숨은 가장 큰 팀의 인원수와 같다. 탈락하면 팀 목숨 -1, 대기열의 다음 사람이 나온다. 인원이 적은 팀은 대기열이 처음으로 순환한다. 필드 선수의 접속이 끊기면 목숨 차감 없이 다음 사람으로 교체, 접속자가 없는 팀은 탈락.
- 필요(What): 팀별 대기열, 목숨 카운터, 접속 끊김 훅.
- 작업 방법: 규칙 단위 테스트 → 봇 시뮬레이션.
- 완료 기준: 필드 팀별 1명, 공통 목숨, 대기열 순환, 접속 끊김 처리.
- 리스크/대안: 경기 중 입장은 관전자로만 받는다.
- 근거: [근거: `IMPLEMENTATION_PLAN.md` 2.5, 2.6, `HANDOFF.md` T-031]

3.5 방 설정과 흐름

- 의도(Why): 친구끼리 즉석에서 모드와 팀을 정해 시작할 수 있어야 한다. 서버는 잘못된 설정을 거부한다.
- 설계(How): 설정(방장 전용): 방 이름 1~24자, 최대 인원 2~22, 모드 6종, 팀 수(개인전 0, 팀전 2~5), 아군 공격 on/off, 제한 시간/목표 킬(데스매치 계열), 맵. 시작 조건은 2명 이상이며 팀 모드는 인원이 있는 팀이 2개 이상. 경기 중 입장은 데스매치 계열과 점령전만 허용, 서바이벌 계열과 태그매치는 관전. 방장이 나가면 가장 오래된 참가자가 승계, 빈 방은 즉시 삭제, 동시 방 상한 기본 8개. 한 플레이어는 동시에 두 방에 들어갈 수 없다. 목표 킬은 데스매치 계열에서만 표시(D19).
- 필요(What): `RoomManager`, `room-rules.ts` 검증 함수, 대기실 UI.
- 작업 방법: 서버 테스트로 이중 입장 거부, 잘못된 모드/팀 조합 거부 확인.
- 완료 기준: 방장만 설정을 바꾸고 잘못된 조합을 서버가 거부한다.
- 리스크/대안: 같은 브라우저 탭 두 개는 같은 사람으로 대체 처리되어 2인 시험은 시크릿 창 필요.
- 근거: [근거: `IMPLEMENTATION_PLAN.md` 2.6, `HANDOFF.md` T-016~T-018, `room-rules.ts` 검증]

3.6 맵 데이터와 품질 조건

- 의도(Why): 서버 충돌과 렌더가 같은 데이터에서 나와 어긋나지 않게 한다. 벽과 낭떠러지가 섞여야 반사 전략과 장외 위험이 동시에 생긴다.
- 설계(How): `maps/mare-01.json`에 `seed`, `playRadius`(60), `bounds`(killY -12, killRadius 110, killTop 60), `terrain`(size 140, resolution 128, amplitude 2.5, 가장자리 감쇠 시작 52/폭 10/깊이 40, craters), `solids`, `spawns`, `conquest`(ringRadius 46, 중립 A/B/C, 주요 반경 7)를 둔다(예시값, 계획서 6.3). 지형 노이즈는 정수 해시로 서버/브라우저가 같은 값을 얻는다. 품질 조건: 반사를 쓸 벽/구조물, 둘레 일부는 벽/일부는 낭떠러지, 스폰은 장외 경계에서 25m 이상, 스폰 24개 이상. 이후 `mare-01-bumpers`(하늘에 판 4개, 맞으면 판에서 13m/s로 튕김)와 경기장 아래를 윗면이 잘린 구(반지름 70)로 만드는 `terrain.dome`이 추가됐다.
- 필요(What): 시드 지형 생성, 삼각형 격자, 맵 검증 테스트(`map.test.ts`).
- 작업 방법: JSON 로더 → 지형 → 구조물 → 스폰 검증 테스트.
- 완료 기준: 맵이 품질 조건을 만족하고 같은 시드가 항상 같은 삼각형을 만든다.
- 리스크/대안: 맵이 1종이라 단조롭다 → 반사판 맵으로 확장됨.
- 근거: [근거: `IMPLEMENTATION_PLAN.md` 6.3, 4.5, `HANDOFF.md` T-003, T-035, 로그 11-08]

3.7 봇과 이스터에그

- 의도(Why): 혼자서도 테스트/플레이할 수 있게 하고, 첫 버전 이후 놀이 요소를 더한다.
- 설계(How): 방 설정 botCount, 서버 AI(sim/bot.ts). 소켓 없이 서버에서 입력을 만들며 사람이 들어오면 대기 중에 자리를 내준다. 로비의 "혼자 봇전 하기"는 봇 5명. 이스터에그: 벽/사람에 막히지 않고 지구 방향 사거리 끝까지 간 충격파를 서버가 세어 경기 인원(봇 포함) x 5발이면 지구가 날아가고 새 경기에서 복구된다(sim/earth.ts).
- 필요(What): 봇 입력 생성, 카운터, 지구 이펙트.
- 작업 방법: 서버 시뮬레이션 테스트(earth.test.ts).
- 완료 기준: 봇전이 한 판 진행되고 이스터에그 카운트가 작동.
- 리스크/대안: 봇은 첫 계획서에 없던 사용자 요청 추가 기능이다(범위 밖 항목이 아님).
- 근거: [근거: HANDOFF.md 로그 11-08, earth.ts 존재]

3. 데이터/리소스 명세

테이블	필드	예시/범위
방 설정	name(문자열 1~24), maxPlayers(정수 2~22), mode, teamCount(0	2~5), friendlyFire, timeLimit, killTarget, mapId, botCount(0~maxPlayers-1)
팀	id, nameKo, nameEn, color, mark	1 빨강 #E5484D ●, 2 파랑 #3E8EF7 ▲, 3 노랑 #F5C518 ■, 4 초록 #30A46C ◆, 5 보라 #8E4EC6 ★, 개인전 색 #F2F4F7
거점	id, pos, radius, owner, progress[team]	주요 반경 7, 중립 반경 6
맵	6.3 JSON 스키마	mare-01, mare-01-bumpers

[근거: teams.ts, IMPLEMENTATION_PLAN.md 6.3, room-rules.ts]

4. 작업 순서

1. 모드 공통 구조와 결과 화면 → 2. 데스매치 계열 → 3. 서바이벌 계열 → 4. 점령전 → 5. 태그매치 → 6. 봇 시뮬레이션 전 모드 → 7. 맵 완성과 성능 측정. [근거: HANDOFF.md T-027~T-035]

5. 검수 기준

- 6개 모드 모두 시뮬레이션에서 종료. 모드별 실제 플레이 완주는 기록이 없어 미확인. [근거: HANDOFF.md 로그 11-07]
- 점령 게이지 단조 증가, 3팀 점령전 속행, 아군 공격 off 시 충격파 통과 테스트. [근거: IMPLEMENTATION_PLAN.md 7장]