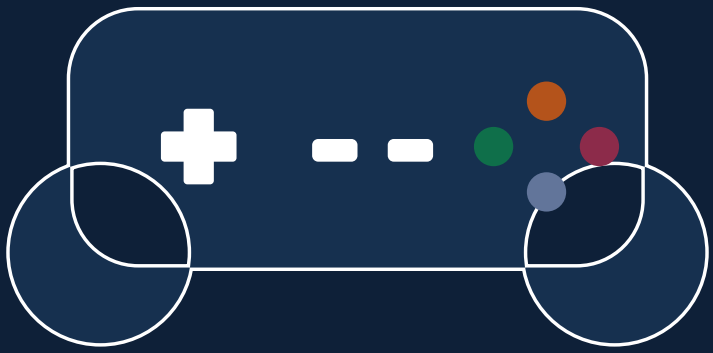
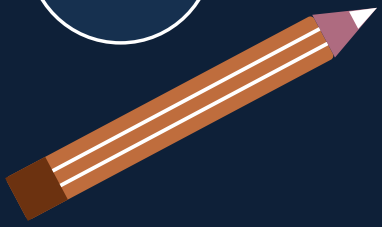
교사를 위한 게임리터러시

AI 에이전트와 함께하는

교실 맞춤형 게임 제작 마스터

게임을 읽는 눈에서
우리 반 게임을 만드는 손까지





이재호·장준형·정지원 지음

감수 김현경·강세진

게임문화재단

게임문화교육원
Game Culture Education Institute

(사)한국창의정보문화학회

교사를 위한 게임리터러시

AI 에이전트와 함께하는 교실 맞춤형 게임 제작 마스터

게임을 읽는 눈에서
우리 반 게임을 만드는 손까지

이재호·장준형·정지원 지음

감수 김현경·강세진

게임문화재단

교사를 위한 게임리터러시

AI 에이전트와 함께하는

교실 맞춤형 게임 제작 마스터

게임을 읽는 눈에서 우리 반 게임을 만드는 손까지

발 행 일	2026년 9월 11일
발 행 처	게임문화재단
발 행 인	유병한
지 은 이	이재호, 장준형, 정지원
감 수	김현경, 강세진
주 관	게임문화교육원
운 영	(사)한국창의정보문화학회
주 소	서울특별시 서초구 방배로 114 다이치빌딩 2층 (게임문화재단)
전 화	02-586-3558
이 메 일	3cgameliteracy@gmail.com
I S B N	979-11-24801-05-5 (13370)

[비매품]

· 이 책의 내용에 대한 저작권은 게임문화재단에 있으며, 2026 교사 게임리터러시 사업의 재정 지원에 의해 제작된 것입니다.

· 이 책을 게임리터러시 관련 교육 및 연수에 사용하시는 경우 출처를 명시하여야 합니다.

· 이 책의 내용을 무단으로 복제 또는 전재하는 것은 법적으로 금합니다.

화면 갈무리 고지. 본문의 화면 갈무리는 2026년 8월 기준이며, 서비스 정책에 따라 버튼의 위치와 이름이 달라질 수 있습니다. 개인정보에 해당하는 이메일 주소는 흐리게 처리했습니다.

상표 고지. Google, Google AI Studio, Gemini, Cloud Run, Firebase, Firestore는 Google LLC의 상표이며, GitHub 및 GitHub Pages는 GitHub, Inc.의 상표입니다. 본 교재는 해당 기업이 후원하거나 승인한 자료가 아닙니다.

서체 고지. 본문은 본명조(Noto Serif CJK KR), 제목과 화면 명칭은 본고딕(Noto Sans CJK KR)을 사용했습니다. 두 서체 모두 SIL Open Font License를 따릅니다.

게임문화재단 · 게임문화교육원 Game Culture Education Institute · (사)한국창의정보문화학회

머리말

이 책은 교사가 게임을 비판적으로 읽고, AI 에이전트로 학습 게임을 직접 만들어, 학생에게 배포하기까지의 전 과정을 담았습니다. 코딩을 배운 적이 없어도 됩니다. 필요한 것은 가르치려는 내용에 대한 선생님의 전문성과, 화면을 따라가 볼 마음뿐입니다.

학생들에게 게임은 여가 중 하나가 아니라 생활 공간 그 자체입니다. 전 국민의 게임 이용률이 50.2%인 데 비해 10대는 80.9%에 이릅니다. 그 공간을 금지의 대상으로만 다루면 교사는 학생의 세계 밖에 남게 됩니다. 이 책은 다른 길을 제안합니다. 게임의 규칙을 읽어 내고, 그 규칙에 학습 목표를 실어, 우리 반에 필요한 게임을 교사가 직접 만드는 길입니다.

몇 해 전이라면 무리한 제안이었을 것입니다. 지금은 다릅니다. AI 에이전트에게 한국어로 설명하면 작동하는 웹 게임이 만들어집니다. 교사가 할 일은 코드를 쓰는 것이 아니라 **무엇을 만들지 정확히 말하는 것**입니다. 그것은 원래 교사가 가장 잘하는 일입니다.

이 책의 구조—여섯 개의 단원

차시가 아니라 주제를 기준으로 여섯 단원을 두었습니다. 각 단원이 독립적으로 완결되므로, 연수 시간과 참여자 수준에 따라 순서와 배분을 조정할 수 있습니다.

단원	제목	다루는 것
1	게임리터러시의 이해	게임을 읽는 눈—MDA ·GBL ·비판적 읽기
2	AI 에이전트와 게임 개발	에이전트의 원리와 바이브 코딩
3	AI Studio 첫걸음	화면 탐색과 첫 퀴즈 게임 제작
4	교과 맞춤 학습 게임	내 교과에 맞춘 게임과 재미 장치 강화
5	온라인 출판	GitHub 저장소와 Pages로 주소 만들기
6	실시간 리더보드	Firebase 연동과 학급 순위표 구현

1–2단원이 이론, 3–6단원이 실습입니다. 실습편은 전부 Google AI Studio를 씁니다.

책을 덮을 때 남는 것

이 과정을 마치면 교사 한 사람이 세 가지를 갖게 됩니다. 첫째, 교과 내용을 담은 학습 웹 게임. 둘째, 학생이 링크나 QR로 접속하는 배포본. 셋째, 우리 반의 기록이 실시간으로 보이는 리더보드까지 붙은 완성작입니다.

읽는 방법

실습편의 화면 갈무리는 실제 화면을 촬영한 것입니다. 화면을 옆에 두고 그대로 따라오시면 됩니다. 다만 서비스는 계속 바뀝니다. 그림과 화면이 다르면 **같은 뜻의 메뉴**를 찾으세요. 버튼의 위치는 바뀌어도 하는 일은 잘 바뀌지 않습니다.

각 단원 끝에는 100분 안팎의 미션과 자주 막히는 지점을 모은 문답이 있습니다. 연수로 운영한다면 5단원과 6단원은 각각 두 차시로 나누기를 권합니다.

> **실습에 필요한 준비물**
> 인터넷이 되는 노트북 한 대, Google 계정, 그리고 GitHub 계정(5단원부터). 학교 망에서 막히는 경우가 있으니, 연수 전날 실제 교실 기기 한 대로 접속을 확인해 두시면 좋습니다.

차례

1단원 게임리터러시의 이해 1

1.1 왜 게임인가—학생들이 사는 세계 3
1.2 게임리터러시란 무엇인가 5
1.3 세 층위—읽기, 해석, 창작 7
1.4 이론적 뿌리—놀이를 사유한 두 고전 9
1.5 게임 분석의 렌즈—MDA 프레임워크 11
1.6 MDA로 게임 읽기—함께 해보는 예시 13
1.7 재미에도 종류가 있다—미학의 8분류 14
1.8 활동—내가 아는 게임을 해부하라 16
1.9 헷갈리는 두 개념—GBL과 게이미피케이션 17
1.10 교실 속 게임 활용—세 갈래 길 22
1.11 비판적으로 읽기—게임이 나를 붙잡는 장치들 24
1.12 소비자에서 생산자로—왜 교사가 만들어야 하는가 26
1.13 함께 가는 길—게임리터러시 지원 생태계 27
1.14 단원 마무리 29

2단원 AI 에이전트와 게임 개발 33

2.1 AI 에이전트란—챗봇과 무엇이 다른가 35
2.2 에이전트는 어떻게 일하는가—다섯 단계 루프 37
2.3 교육 현장의 AI 에이전트—이미 시작된 변화 38
2.4 바이브 코딩—대화가 곧 개발이다 40
2.5 역할 분담—교사의 전문성이 곧 개발력 41
2.6 바이브 코딩 도구 세 가지 42
2.7 왜 Google AI Studio인가—네 가지 결정타 43
2.8 AI Studio 기능 지도—앞으로 만날 여섯 무기 44
2.9 프롬프트 한 줄이 게임이 되는 순간 45

2.10 실습 로드맵—우리가 만들 것들 46
2.11 시작 전 안전 수칙—네 개의 방패 47
2.12 실습 준비 체크리스트 47
2.13 단원 마무리 48

3단원 AI Studio 첫걸음 51

3.1 오늘의 퀘스트와 실습 여정 53
3.2 접속과 로그인 55
3.3 빌드 홈 투어—다섯 구역만 알면 됩니다 57
3.4 입력창—게임을 시작하는 세 가지 방법 60
3.5 실습 프롬프트와 실행 62
3.6 생성 관찰—에이전트가 일하는 2–3분 63
3.7 완성—첫 게임을 플레이하고 점검하기 65
3.8 대화로 수정하기—개발자와 회의하듯 66
3.9 주석 모드—말 대신 화면을 짚어서 68
3.10 Code 탭 구경과 이름 붙이기 70
3.11 20분 미션—내 교과의 OX 퀴즈로 71
3.12 트러블슈팅—막히는 여섯 지점 72
3.13 단원 마무리 73

4단원 교과 맞춤 학습 게임 75

4.1 이 단원의 퀘스트와 여정 77
4.2 미니 기획서—학습 목표를 게임 규칙으로 번역하기 79
4.3 장르 카탈로그—교과가 형식을 고른다 81
4.4 본격 게임 생성—기획서를 프롬프트로 83
4.5 게임 요소 강화—재미 장치 고르고 붙이기 86
4.6 난이도 곡선·배려 설계와 AI 실시간 기능 89
4.7 플레이테스트—두 개의 시점으로 검증하기 91
4.8 60분 미션—내 수업의 본격 게임을 완성하라 92

4.9 교실 적용—언제, 어떻게 쓰는 게임인가 94
4.10 자주 막히는 지점 여섯 95
4.11 단원 마무리 96

5단원 온라인 출판 99

5.1 왜“파일”이 아니라“주소”여야 할까 101
5.2 출판 방법 네 가지 102
5.3 갈림길—딱 하나만 물어보세요 103
5.4 공유 링크 만들기 104
5.5 Cloud Run에 게시하기 106
5.6 GitHub 기초—교사에게 필요한 개념 넷 108
5.7 GitHub 화면 익히기 109
5.8 저장소는 누가 만드는가—두 갈래 112
5.9 AI Studio에서 코드 올리기 112
5.10 올라갔는지 눈으로 확인하기 116
5.11 Pages 켜기—주소가 태어나는 곳 119
5.12 학생에게 닿게 하기 123
5.13 수업 배치—이 게임을 언제 꺼내는가 125
5.14 고치고 다시 올리기, 그리고 회수하기 126
5.15 공개하기 전 3분 점검 127
5.16 100분 미션—내 게임에 주소 달기 128
5.17 자주 막히는 지점 여덟 130
5.18 단원 마무리 131

6단원 실시간 리더보드 135

6.1 순위표는 약인가 독인가—만들기 전에 정할 것 139
6.2 길이 바뀝니다—이제 서버가 필요합니다 140
6.3 Firestore —교실 뒤에 걸린 클라우드 칠판 141
6.4 Firebase 콘솔에서 프로젝트 만들기 143

6.5 Firestore 열기—컬렉션·문서·필드 만들기 149
6.6 우리 리더보드의 데이터 설계 153
6.7 로그인을 걷어냅니다—초등 교실의 핵심 갈림길 154
6.8 익명 인증 켜기—콘솔에서 스위치 하나 155
6.9 보안 규칙—지금 내 데이터베이스는 안전한가 160
6.10 규칙 고쳐 쓰고 시험해 보기 162
6.11 앱에 붙이기—Enable Firebase 165
6.12 리더보드 구현—원칙을 프롬프트로 옮기기 167
6.13 교사용 회차 초기화—지우지 않고 새 판을 엽니다 169
6.14 사용량과 무료 한도—우리 학교는 몇 반까지 170
6.15 학생에게 열기 전 여섯 칸 172
6.16 100분 미션—우리 반 순위표 완성하기 173
6.17 자주 막히는 지점 여덟 174
6.18 단원 마무리 175

WORLD 1

게임리터러시의 이해

게임을 읽고, 해석하고, 만드는 교사의 눈

게임은 학생들이 또래 관계와 성취와 이야기를 먼저 만나는 생활 공간이자, 규칙과 서사와 상호작용이 얽힌 하나의 복합 텍스트입니다.

이 단원에서는 그 텍스트를 읽는 언어를 익힙니다. 호이징가와 카이와의 놀이 이론에서 출발해 게임 분석의 표준 틀인 MDA를 익히고, 게임기반학습과 게이미피케이션을 구분하며, 보상 루프와 확률형 아이템 같은 상업적 설계를 해부합니다.

여기서 익힌 언어가 3단원부터 게임을 직접 만들 때의 기획 언어가 됩니다.

이 단원은 실습이 없는 유일한 단원입니다. 손으로 무엇을 만들지는 않지만, 뒤이어 만들 것의 설계도를 그리는 시간입니다. 3단원부터 선생님은 AI에게 게임을 설명하게 됩니다. 그때 "재미있게 만들어 줘"라고 말하는 교사와 "도전과 발견을 축으로, 실패해도 즉시 재시도할 수 있는 규칙으로 만들어 줘"라고 말하는 교사의 결과물은 다릅니다. 그 차이를 만드는 것이 이 단원의 어휘입니다.

내용은 이론이지만 서술은 교실 쪽을 봅니다. 본문은 "이 개념이 우리 반에서 무엇으로 보이는가"를 중심에 두고, 근거가 되는 문헌은 그 자리 아래쪽 각주에 APA 형식으로 달았습니다. 단원 말미에는 같은 문헌을 한 목록으로 다시 모아 두었습니다.

> **이 단원을 마치면**
>
> 게임리터러시의 개념과 세 층위를 설명할 수 있습니다. 임의의 게임을 MDA 세 겹으로 분해할 수 있습니다. 게임기반학습과 게이미피케이션을 구분해 내 수업의 적용점을 찾을 수 있습니다. 그리고 게임의 설득 구조를 금지가 아닌 해부의 관점에서 학생과 함께 읽어낼 수 있습니다.

시작 문장은 이 책 전체를 통틀어 가장 오래된 인용입니다. "놀이는 문화보다 오래되었다." 요한 호이징가가 1938년『호모 루덴스』의 첫머리에 적은 문장입니다.[1] 게임을 수업에 들이는 일이 새로운 유행처럼 보이더라도, 그 뿌리는 인간이 문화를 만들기 시작한 시점까지 내려갑니다.

1.1 왜 게임인가—학생들이 사는 세계

게임 교육의 출발점은 교사의 취향이 아니라 통계입니다. 게임을 좋아하는 교사든 한 번도 해 본 적 없는 교사든, 우리가 마주하는 학생들의 조건은 같습니다. 그 조건을 숫자로 먼저 확인하고 시작합니다.

한국콘텐츠진흥원이 전국 10–69세 일반 국민 1만 명을 대상으로 조사한 결과, 전 국민의 게임 이용률은 50.2%였습니다. 국민의 절반이 게임을 한다는 뜻이니 그 자체로도 적지 않은 숫자입니다. 그런데 같은 조사에서 10대의 이용률은 80.9%였습니다. 전체 이용률의 약 1.6배이고, 열 명 중 여덟 명입니다. 전체 이용률이 3

[1] Huizinga, J. (1955). *Homo ludens: A study of the play-element in culture.* Beacon Press. (원저 1938년 출판)

년 연속 하락하는 흐름 속에서도 10대만은 압도적 이용층으로 남아 있습니다.[2]

> **숫자 두 개만 기억하면 됩니다**
> 전 국민 50.2% / 10대 80.9%. 학부모 연수나 동료 협의회에서 "왜 게임을 수업에서 다루느냐"는 질문을 받았을 때, 이 두 숫자가 가장 짧은 대답이 됩니다.

성인에게는 여가, 학생에게는 생활 공간

같은 활동이라도 차지하는 자리가 다릅니다. 성인에게 게임은 여러 여가 중 하나입니다. 퇴근 후 30분, 주말 한때의 선택지이고, 그 자리를 산책이나 드라마가 대신해도 삶은 크게 달라지지 않습니다. 학생에게는 그렇지 않습니다. 학생에게 게임은 또래 관계가 맺어지고, 성취 경험이 쌓이고, 이야기가 소비되는 생활 공간 그 자체입니다. 방과 후에 어디서 만나느냐고 물으면 장소가 아니라 게임 이름이 돌아오는 경우를 선생님도 겪어 보셨을 것입니다.

게임은 학생들의 제1언어

학생들은 친교와 성취, 서사와 경쟁이라는 인간의 근본 경험을 게임 안에서 먼저 만납니다. 협력이 무엇인지, 규칙을 지킨다는 것이 무엇인지, 진 다음에 다시 도전한다는 것이 어떤 기분인지를 교실보다 게임에서 먼저 배웁니다. 그래서 게임 용어와 문법은 이미 또래 문화의 공용어입니다. 레벨, 리스폰, 파밍, 쿨타임 같은 말이 교실 대화에 그대로 등장하고, 학생들은 그 말을 설명 없이 주고받습니다.

언어를 모르면 세계를 못 읽는다

학생이 어떤 게임을 왜 좋아하는지 읽지 못하는 교사는 학생 문화의 절반을 놓칩니다. 이것은 수업의 문제만이 아닙니다. 게임 이해는 생활지도와 상담의 자원이기도 합니다. 어떤 게임에서 어떤 역할을 맡는지, 누구와 함께 하는지, 무엇 때문에 다투었는지를 알아들을 수 있는 교사는 상담의 첫 문장을 다르게 시작할 수 있습니다. 반대로 게임 이야기가 나오는 순간 대화가 끊기는 교사에게는 학생이 그 이야기를 다시 꺼내지 않습니다.

[2] 한국콘텐츠진흥원. (2025). *2025 게임이용자 실태조사*. 한국콘텐츠진흥원.

읽기를 넘어 쓰기로

여기까지가 이해의 영역이라면, 이 책이 겨냥하는 곳은 그다음입니다. 학생들이 이미 유창하게 쓰는 이 언어를 수업의 도구로 바꾸는 것이 게임리터러시 교육의 목표입니다. 그리고 그 마지막 단계는 교사가 직접 게임을 만드는 단계입니다. 이 교재는 거기까지 안내합니다. 왜 마지막 단계여야 하는지는 26쪽에서 다시 다룹니다.

1.2 게임리터러시란 무엇인가

리터러시(literacy)는 본래 글을 읽고 쓰는 능력을 가리켰습니다. 그러나 이 말은 미디어 환경이 바뀔 때마다 대상을 넓혀 왔습니다. 새로운 텍스트가 등장하면, 그 텍스트를 읽고 쓰는 능력에 이름이 붙었습니다. 게임리터러시는 그 확장의 가장 최근 지점입니다.

게임 연구자 James Paul Gee는 게임이 고유한 기호 체계와 학습 원리를 지닌 텍스트이며, 따라서 읽는 법과 쓰는 법을 배워야 할 대상이라고 주장했습니다. 게임을 "해도 되는 것"과 "하면 안 되는 것"의 문제로만 다루던 논의를, 텍스트를 다루는 방식의 문제로 옮겨 놓은 관점입니다.[3]

> **정의**
> **게임리터러시(Game Literacy)**란, 게임을 비판적으로 이해하고, 그 의미와 구조를 해석하며, 자신의 목적에 맞게 만들어 표현할 수 있는 능력을 말합니다.

이 정의에는 세 개의 동사가 들어 있습니다. 이해하고, 해석하며, 만든다. 뒤에서 보게 될 세 층위가 이미 이 한 문장에 압축되어 있습니다.

[3] Gee, J. P. (2003). *What video games have to teach us about learning and literacy.* Palgrave Macmillan.

리터러시 개념은 어떻게 넓어져 왔나

문자에서 게임까지, 확장의 경로는 다음과 같습니다. 각 단계는 앞 단계를 대체하는 것이 아니라 그 위에 얹힙니다.[4]

리터러시	다루는 텍스트	핵심 능력
문자 리터러시	글	글을 읽고 쓰는 전통적 문해력
미디어 리터러시	TV·신문 등 대중매체	매체의 메시지를 비판적으로 수용
디지털 리터러시	디지털 도구와 정보	정보를 탐색·평가·활용
게임 리터러시	규칙·서사·상호작용의 복합 텍스트	그 복합 텍스트를 읽고 쓰기

게임이 가장 복합적인 텍스트인 이유

게임은 글과 그림과 소리에 규칙과 상호작용이 결합된 텍스트입니다. 소설은 읽는 사람이 바뀌어도 문장이 바뀌지 않지만, 게임은 플레이하는 사람에 따라 매번 다른 텍스트가 됩니다. 같은 게임을 두 학생이 플레이하면 두 개의 서로 다른 이야기가 생깁니다. 읽는 행위 자체가 텍스트를 생성하는 셈입니다.[5]

그러나 복합적이라는 것이 무질서하다는 뜻은 아닙니다. 문자 텍스트에 문법이 있듯 게임에도 문법이 있고, 그 문법을 아는 사람만이 읽기를 넘어 쓰기, 곧 제작에 이를 수 있습니다. 이 단원의 나머지 절들은 그 문법의 목록입니다. 놀이의 네 가지 얼굴(9쪽), 게임을 세 겹으로 분해하는 MDA(11쪽), 재미의 여덟 가지 이름(14쪽)이 차례로 나옵니다.

> **학생에게 설명할 때**
> "게임도 책처럼 읽는 법이 있다"는 한 문장이면 충분합니다. 책에 줄거리와 문체와 주제가 있듯 게임에는 규칙과 조작감과 재미가 있고, 그것을 말로 설명할 수 있는

[4] Buckingham, D. (2003). *Media education: Literacy, learning and contemporary culture.* Polity Press.; Gilster, P. (1997). *Digital literacy.* Wiley.

[5] 글·그림·소리·규칙이 겹친 텍스트를 다루는 문해력이라는 관점은 복합양식 문해(multiliteracies) 논의에서 왔습니다. New London Group. (1996). A pedagogy of multiliteracies: Designing social futures. *Harvard Educational Review, 66*(1), 60–92.; Buckingham, D., & Burn, A. (2007). Game literacy in theory and practice. *Journal of Educational Multimedia and Hypermedia, 16*(3), 323–349.

> 사람이 게임을 잘 읽는 사람이라고 알려 주세요.

1.3 세 층위—읽기, 해석, 창작

게임리터러시는 세 층위가 차례로 쌓이는 능력입니다. 문자 문해력이 "읽기 → 독해 → 작문"으로 발달하듯, 게임리터러시도 "플레이 → 분석 → 제작"의 계단을 오릅니다. 이 교재의 목표 지점은 가장 높은 층위인 창작입니다.[6]

계단이라는 비유가 중요합니다. 세 층위는 취향의 문제가 아니라 순서의 문제입니다. 플레이해 보지 않은 게임을 분석할 수 없고, 분석해 보지 않은 사람이 설계할 수 없습니다. 자기 반 학생들이 무슨 게임을 하는지 모르면서 학습 게임을 만들면, 만드는 사람만 재미있는 게임이 나옵니다.

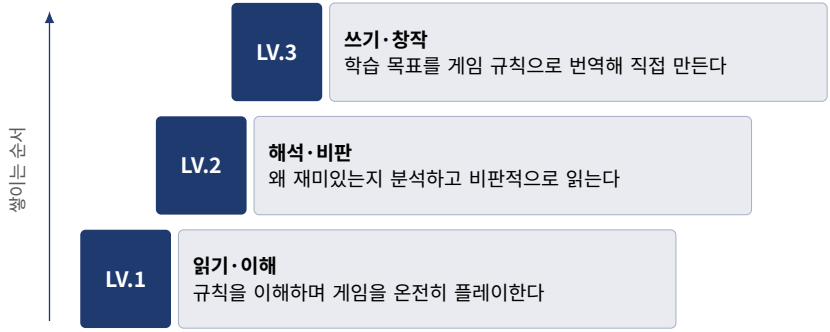


그림 1-1. 게임리터러시의 세 층위. 아래 칸을 밟아야 위 칸으로 올라간다

[6] 읽기와 쓰기를 함께 놓는 게임리터러시 구도는 다음 논의를 따랐습니다. Buckingham, D., & Burn, A. (2007). Game literacy in theory and practice. *Journal of Educational Multimedia and Hypermedia, 16*(3), 323–349.; Gee, J. P. (2003). *What video games have to teach us about learning and literacy.* Palgrave Macmillan. 세 층위의 이름과 구분은 이 책이 연수용으로 다시 정리한 것입니다.

LV.1 읽기 · 이해—온전히 플레이하기

규칙과 장르 관습, 조작 체계를 이해하며 게임을 온전히 플레이하는 단계입니다. 이 단계를 "그냥 노는 것"으로 여기기 쉽지만, 실제로 일어나는 일은 문해 활동에 가깝습니다. 튜토리얼을 읽어내고, 화면에 흩어진 정보에서 목표를 파악하고, 실패한 원인을 추론해 전략을 수정하는 과정 자체가 하나의 읽기입니다.

교사에게 이 단계는 "학생이 하는 게임을 한 번은 직접 해 보기"를 뜻합니다. 영상으로 보는 것과 손으로 해 보는 것은 다릅니다. 30분만 해 보아도 그 게임이 무엇을 요구하는지, 왜 학생이 그만두지 못하는지가 몸으로 이해됩니다.

LV.2 해석 · 비판—왜 재미있는지 말로 설명하기

왜 재미있는지, 어떤 장치가 이용자를 붙잡는지 분석하고 평가하는 단계입니다. MDA 같은 분석 틀로 재미의 구조를 언어화하고, 확률형 아이템 등 상업적 설계를 비판적으로 검토합니다. "이 게임 재미있어요"와 "이 게임은 실패 후 재시도까지 3초밖에 안 걸려서 계속 하게 돼요" 사이의 거리가 곧 LV.1과 LV.2의 거리입니다.

이 단계는 학생과 함께 오를 수 있습니다. 아니, 함께 오르는 편이 낫습니다. 학생은 LV.1의 경험이 교사보다 풍부하므로, 분석의 재료를 학생이 대고 언어를 교사가 대는 협업이 가능합니다.

LV.3 쓰기 · 창작—학습 목표를 규칙으로 번역하기

자신의 목적에 맞는 게임을 직접 만드는 단계입니다. 교사에게 그 목적은 수업입니다. 학습 목표를 게임 규칙으로 번역하는 이 단계가 게임리터러시의 완성이며, 3–6단원 실습의 내용입니다.

"번역"이라는 말을 눈여겨보시기 바랍니다. 분수의 크기를 비교할 수 있다는 목표를 그대로 화면에 띄우면 문제집이 되지만, 큰 분수가 적힌 물고기를 낚아야 점수가 오르는 규칙으로 바꾸면 게임이 됩니다. 목표는 같고 형식이 다릅니다. 이 번역이 교사가 하는 설계 작업의 핵심입니다.

> **가로막고 있던 것**
>
> 과거에는 코딩이 LV.3으로 가는 계단을 가로막았습니다. 아이디어가 있어도 구현할 수단이 없으면 창작은 시작되지 않습니다. AI 에이전트가 그 장벽을 없앴고, 그 이야기를 2단원에서 다룹니다. 이 교재의 종착점은 LV.3입니다.

1.4 이론적 뿌리—놀이를 사유한 두 고전

게임 연구의 뿌리에는 두 권의 고전이 있습니다. 두 학자 모두 놀이를 시간 낭비가 아닌 인간 문명의 근본 활동으로 격상시켰고, 오늘날 게임 교육을 정당화하는 이론적 토대가 되었습니다. 학교에서 게임을 다루는 일이 왜 정당한지 설명해야 할 때, 이 두 이름이 가장 든든한 근거입니다.

요한 호이징가, 『호모 루덴스』(1938)

호이징가의 명제는 한 줄로 요약됩니다. 인간은 놀이하는 존재, 곧 호모 루덴스(Homo Ludens)라는 것입니다. 그가 뒤집은 것은 순서입니다. 흔히 문화가 먼저 있고 그 여유에서 놀이가 나온다고 생각하지만, 호이징가는 놀이가 문화를 만들어 낸 원천이라고 보았습니다. 법정의 공방, 예술의 형식, 학문의 논쟁 모두 그 뿌리에 규칙 있는 겨루기, 곧 놀이가 있습니다. 재판이 정해진 절차와 역할과 승패를 갖춘 활동이라는 점을 떠올려 보면 이 주장이 무리하지 않게 들립니다.[7]

마법의 원—교실 게임이 안전한 이유

호이징가가 남긴 또 하나의 개념이 교사에게 특히 유용합니다. 놀이는 일상과 구분되는 시공간, 이른바 **마법의 원(magic circle)** 안에서 고유한 규칙으로 진행된다는 것입니다. 원 안에서는 원 밖의 규칙이 잠시 멈춥니다. 술래에게 붙잡히는 일이 실제 체포가 아니듯, 게임 속 실패는 실제 실패가 아닙니다.[8]

게임 속 실패가 안전한 이유가 여기에 있고, 교실 게임이 "안전한 도전"을 제공할 수 있는 근거도 여기에 있습니다. 평소 손을 들지 않는 학생이 게임에서는 세 번 틀리고 네 번째에 맞히는 장면을 보신 적이 있을 것입니다. 원 안에서는 틀리는 일의 값이 싸기 때문입니다. 다만 이 원은 교사가 깨뜨릴 수도 있습니다. 게임의 점수를 성적이나 상점에 연결하는 순간 원 밖의 규칙이 원 안으로 들어오고, 실패는 다시 비싸집니다.

[7] Huizinga, J. (1955). *Homo ludens: A study of the play-element in culture.* Beacon Press. (원저 1938년 출판)

[8] "마법의 원"이라는 표현은 호이징가에게서 나왔고, 오늘날의 게임 설계 용어로 다듬은 것은 다음 책입니다. Salen, K., & Zimmerman, E. (2004). *Rules of play: Game design fundamentals.* MIT Press.

> **수업 함의**
> 게임은 시간 낭비가 아니라 인간 학습의 원형입니다. 그리고 게임을 학습에 쓸 때의 가장 큰 자산은 재미가 아니라 "실패해도 괜찮은 공간"이라는 성질입니다.

로제 카이와, 『놀이와 인간』(1958)

카이와는 호이징가의 논의를 이어받아 놀이를 네 갈래로 분류했습니다. 이 분류는 학습 게임을 설계할 때 재료 목록처럼 쓸 수 있어 실용적입니다.[9] 지금 만들려는 게임이 어떤 종류의 놀이에 기대고 있는지를 확인하는 점검표로 삼으시면 됩니다.

유형	어떤 놀이인가	예
아곤(경쟁)	실력을 겨루어 승부를 가림	달리기 시합, 바둑, 대전 게임, 퀴즈 대결
알레아(운)	결과를 우연에 맡김	주사위, 제비뽑기, 카드 뽑기
미미크리(모방)	다른 존재가 되어 봄	역할놀이, 연극, RPG, 아바타 꾸미기
일링크스(현기증)	감각의 혼란 자체를 즐김	회전목마, 롤러코스터, 스피드 질주

섞을 때 생기는 힘

좋은 학습 게임은 대개 두 가지 이상의 놀이 요소를 섞습니다. 예컨대 퀴즈 대결(아곤)에 뽑기 보상(알레아)을 더하면 실력 차가 있어도 모두가 끝까지 참여합니다. 순수한 아곤만으로 운영되는 게임은 상위권 몇 명이 승패를 결정한 뒤 나머지 학생이 이탈합니다. 여기에 우연이 조금 섞이면 마지막 문항까지 결과를 알 수 없게 되고, 참여가 유지됩니다.

교실에서 흔히 쓰는 조합을 카이와의 말로 옮겨 보면 자신이 이미 무엇을 하고 있었는지 보입니다. 모둠별 골든벨은 아곤이고, 뽑기 상자로 순서를 정하는 것은 알레아이며, 역사 인물이 되어 발표하는 활동은 미미크리입니다. 일링크스는 화면 안에서 속도감으로 구현되며, 3단원 이후 만들 게임에서 연출의 재료가 됩니다.

[9] Caillois, R. (2001). *Man, play and games* (M. Barash, Trans.). University of Illinois Press. (원저 1958년 출판)

1.5 게임 분석의 렌즈—MDA 프레임워크

MDA는 게임 연구자 Hunicke, LeBlanc, Zubek이 2004년 제안한 게임 분석·설계의 표준 틀입니다.[10] 게임을 메커닉스(규칙) – 다이내믹스(상호작용) – 미학(재미 경험)의 세 겹으로 나누어 봅니다. 복잡해 보이는 게임도 이 세 겹으로 분해하면 구조가 선명하게 드러납니다.

이 단원에서 딱 하나만 가져가야 한다면 MDA입니다. 나머지 개념은 배경 지식에 가깝지만, MDA는 3단원 이후 매번 손에 들고 쓰게 되는 도구입니다. AI에게 게임을 설명할 때도 이 세 글자의 순서대로 말하면 요구사항이 정리됩니다.

M · 메커닉스—게임을 구성하는 규칙과 부품

점프, 점수, 타이머, 아이템, 승리 조건처럼 설계자가 직접 정하는 요소들입니다. 게임의 "법전"에 해당합니다. 메커닉스의 특징은 눈에 보이고 셀 수 있다는 점입니다. 목숨이 세 개인지 다섯 개인지, 제한 시간이 60초인지 90초인지는 문서에 그대로 적을 수 있습니다.

교사가 게임을 만들 때 직접 손대는 곳도 여기뿐입니다. 뒤의 두 겹은 만들어지는 것이지 만드는 것이 아닙니다.

D · 다이내믹스—플레이 중 실제로 벌어지는 일

규칙이 플레이어와 만나 만들어내는 행동 패턴입니다. 눈치 싸움, 역전극, 협력과 배신은 규칙에 적혀 있지 않지만 규칙에서 태어납니다. 어느 게임의 규칙서에도 "마지막 문항에서는 모험을 걸어라"라고 쓰여 있지 않지만, 점수 배점을 뒤로 갈수록 크게 하면 그 행동은 저절로 나타납니다.

다이내믹스는 설계자의 의도를 배신하기도 합니다. 협력하라고 만든 규칙이 한 학생에게 일을 몰아주는 결과를 낳는 경우가 대표적입니다. 그래서 만든 게임은 반드시 사람을 앉혀 놓고 돌려 봐야 합니다. 규칙만 읽어서는 무엇이 벌어질지 알 수 없습니다.

[10] Hunicke, R., LeBlanc, M., & Zubek, R. (2004). MDA: A formal approach to game design and game research. In *Proceedings of the AAAI Workshop on Challenges in Game AI*. AAAI Press.

A · 미학—플레이어가 느끼는 재미 경험

성취감, 몰입, 짜릿함, 우정처럼 플레이어의 마음에 남는 정서적 반응입니다. 게임의 최종 목적지입니다. 학생이 수업 후에 기억하는 것도 규칙이 아니라 이 층위입니다.

두 개의 시점—만들 때와 읽을 때

MDA의 진짜 쓸모는 세 글자의 뜻보다 방향에 있습니다. 디자이너는 M에서 출발해 규칙을 설계함으로써 재미(A)를 빚어내고, 플레이어는 A에서 출발해 재미를 먼저 느낀 뒤에야 규칙(M)을 의식합니다. 같은 게임을 두고 두 사람이 정반대 방향으로 걷는 셈입니다.

교사는 이 두 시점을 오가는 사람입니다. 게임을 분석할 때는 플레이어처럼 재미에서 출발해 규칙을 역추적하고, 게임을 만들 때는 디자이너처럼 규칙을 조립해 재미를 겨냥합니다. 이 왕복이 익숙해지면 "학생들이 좋아하는 저 게임의 어떤 규칙을 우리 수업으로 가져올 수 있을까"라는 질문에 답할 수 있게 됩니다.

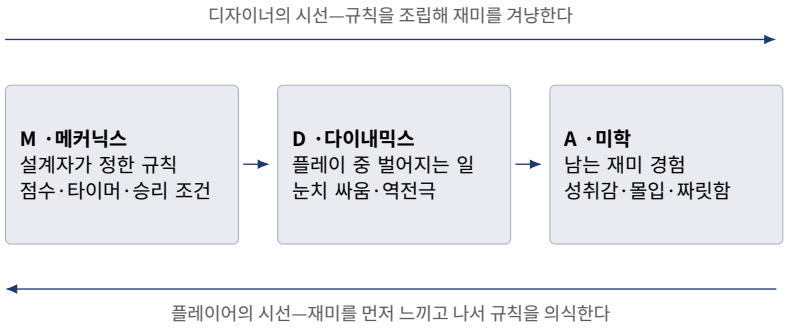


그림 1-2. MDA의 세 겹과 두 개의 시점. 만드는 사람과 노는 사람은 정반대 방향으로 걷는다

> **세 글자를 외우는 방법**
> M은 만든 것, D는 벌어진 일, A는 남은 감정. 이 세 마디로 기억하면 실제 분석에서 헷갈리지 않습니다.

1.6 MDA로 게임 읽기—함께 해보는 예시

개념만으로는 손에 붙지 않습니다. 누구나 아는 달리기 점프 게임(플랫포머)을 MDA로 분해해 봅니다. 아래 표처럼 세 층위를 나누어 적기만 해도, "이 게임의 재미가 어느 규칙에서 나오는지"가 보이기 시작합니다.

층위	묻는 것	달리기 점프 게임의 경우
메커닉스	무엇으로 되어 있나	달리기·점프·장애물·코인·목숨 3개·제한 시간·스테이지 클리어 조건
다이내믹스	무슨 일이 벌어지나	아슬아슬한 연속 점프, 코인 욕심을 내다 낭떠러지로 떨어지는 선택의 긴장, 남은 시간이 적을 때의 전력 질주
미학	무엇을 느끼나	도전(어렵지만 될 것 같다), 성취(드디어 깼다), 감각(속도감과 리듬)

표를 읽는 법

첫 줄의 항목들은 하나하나가 설계자가 정한 규칙이며, 이 목록을 바꾸면 전혀 다른 게임이 됩니다. 목숨을 무한으로 바꾸면 긴장이 사라지고, 제한 시간을 없애면 전력 질주가 사라집니다. 코인을 없애면 낭떠러지 앞에서 망설이는 선택 자체가 없어집니다.

둘째 줄은 어디에도 적혀 있지 않은 일들입니다. 규칙 어디에도 "긴장하라"고 쓰여 있지 않지만, 규칙의 조합이 이 행동들을 만들어냅니다. 특히 눈여겨볼 것은 코인과 낭떠러지의 관계입니다. 코인이라는 보상과 추락이라는 위험이 같은 자리에 놓이는 순간, 플레이어는 매번 선택을 강요받습니다. 이 긴장은 코인 규칙 하나, 낭떠러지 규칙 하나로는 생기지 않고 둘이 만나야만 생깁니다.

셋째 줄이 학생의 마음에 남는 부분입니다. 실패해도 다시 도전하게 만드는 이 감정이 게임이 최종적으로 판매하는 상품입니다. 그리고 학습 게임에서는 이것이 곧 "한 번 더 풀어 보게 만드는 힘"이 됩니다.

핵심 질문 하나

분석의 목적은 표를 채우는 것이 아닙니다. 다음 한 문장을 완성하는 것입니다.

> **핵심 질문**
> "이 게임의 재미(A)는 어떤 규칙(M)에서 나오는가?" 이 연결 고리를 언어로 표현하는 것이 게임 분석의 전부입니다.

달리기 점프 게임이라면 이렇게 답할 수 있습니다. "이 게임의 성취감은 목숨 3개와 스테이지 클리어 조건에서 나온다. 잃을 것이 있고 도달할 지점이 분명하기 때문이다." 문장의 형태가 중요합니다. 재미의 이름을 앞에 두고, 그 재미를 만든 규칙을 뒤에 둡니다. 이 문장을 쓸 수 있으면 같은 구조를 내 수업으로 옮길 수 있습니다.

1.7 재미에도 종류가 있다—미학의 8분류

MDA의 저자들은 "재미"라는 뭉뚱그린 말 대신 여덟 가지 미학을 제시했습니다.[11] 같은 "재미있는 게임"이라도 어떤 재미인지가 다르며, 학습 게임을 설계할 때는 이 중 무엇을 노릴지 의식적으로 선택해야 합니다.

이 목록이 필요한 이유는 실무적입니다. "재미있게 만들어 주세요"라는 요구는 아무것도 지시하지 않습니다. 반면 "발견의 재미를 주세요"라는 요구는 숨겨진 것을 찾는 구조를 만들라는 구체적 지시가 됩니다. 여덟 개의 이름은 그런 지시를 가능하게 하는 어휘입니다.

[11] Hunicke, R., LeBlanc, M., & Zubek, R. (2004). MDA: A formal approach to game design and game research. In *Proceedings of the AAAI Workshop on Challenges in Game AI*. AAAI Press.

미학	어떤 즐거움인가
감각(Sensation)	보고 듣는 즐거움. 색·소리·타격감 등 감각적 만족
환상(Fantasy)	다른 존재가 되는 상상. 역할과 세계관 몰입
서사(Narrative)	이야기를 따라가는 맛. 전개와 결말의 궁금증
도전(Challenge)	한계를 넘는 짜릿함. 학습 게임의 핵심 재미
친교(Fellowship)	함께 노는 즐거움. 협력·경쟁의 사회적 재미
발견(Discovery)	숨겨진 것을 찾는 재미. 탐험과 새로움
표현(Expression)	나를 드러내는 기쁨. 꾸미기·창작·선택
소일(Submission)	가볍게 반복하는 편안함. 부담 없는 루틴

학습 게임의 축은 도전

여덟 가지가 대등한 것은 아닙니다. 학습 게임의 핵심 재미는 도전입니다. 학습이란 본래 지금 할 수 없는 것을 할 수 있게 되는 일이고, 그 과정이 주는 감정이 곧 도전이기 때문입니다. 도전이 빠진 학습 게임은 활동은 있으나 배움이 없는 시간이 되기 쉽습니다.

문제는 도전 하나만으로는 오래 가지 않는다는 점입니다. 어려운 문제만 계속 나오면 상위권만 남습니다. 그래서 잘 만든 학습 게임은 보통 도전을 축으로 2–3가지 재미를 조합합니다. 도전에 친교를 더하면 모둠 퀴즈 대전이 되고, 도전에 발견을 더하면 탐험형 문제 풀이가 됩니다.

설계 질문

만들기 전에 스스로 물어야 할 질문은 이것입니다. 내가 만들 학습 게임은 여덟 가지 중 어떤 재미를 줄 것인가. 답이 셋을 넘어가면 초점이 흐려지고, 하나뿐이면 단조로워집니다. 도전 하나에 다른 하나를 얹는 정도가 첫 게임으로는 적당합니다.

> **학생의 대답이 곧 목록입니다**
>
> “그 게임이 왜 재미있어요?”라는 발문에 학생들이 쏟아내는 대답을 그대로 받아 적으면, 대개 이 여덟 가지 중 몇 개로 정리됩니다. 학생의 말을 이 이름으로 옮겨 주는 것만으로도 분석 수업이 됩니다.

1.8 활동—내가 아는 게임을 해부하라

이론은 직접 적용해 보아야 몸에 붙습니다. 모둠별로 게임 하나를 골라 MDA 세 층위로 분해하고, 재미와 규칙의 연결 고리를 한 문장으로 정리해 발표합니다. 소요 시간은 약 15분이며, 연수에서도 학급에서도 같은 방식으로 진행할 수 있습니다.

1. 모둠별로 모두가 아는 게임 1개를 선정합니다. 학생들이 실제로 즐기는 게임일수록 좋습니다. 교육용 게임을 고르면 분석은 쉬워지지만 대화가 뜨거워지지 않습니다.
2. MDA 세 칸에 나눠 적습니다. 규칙은 무엇인가, 플레이 중 무슨 일이 벌어지는가, 어떤 재미를 느끼는가. 세 칸을 동시에 채우려 하지 말고 반드시 M부터 순서대로 채웁니다.
3. "이 게임의 (A)는 (M)에서 나온다" 형식으로 연결 고리를 한 문장으로 정리합니다. 문장이 안 만들어지면 대개 M을 너무 뭉뚱그려 적은 경우입니다.
4. 모둠별 1분 공유. 같은 게임도 사람마다 다르게 읽힘을 확인하는 것이 이 활동의 숨은 목표입니다.

워크시트—세 칸이면 충분합니다

별도의 양식이 필요하지는 않습니다. 종이 한 장을 세 칸으로 나누고 다음 세 질문만 적어 두면 됩니다.

> **워크시트 미리보기**
> 게임 이름: _____
>
> **M · 규칙과 부품** 어떤 규칙과 요소로 이루어져 있나요?
>
> _____
>
> **D · 벌어지는 일** 플레이 중 어떤 행동·상황이 반복되나요?
>
> _____
>
> **A · 느끼는 재미** 8분류 중 어떤 재미인가요? (복수 선택)
>
> _____

> 한 문장 정리: 이 게임의 _____는(은) _____에서 나온다.

진행할 때 챙길 것

첫째, 정답을 만들지 마세요. 같은 게임을 두고 어떤 모둠은 친교를 적고 어떤 모둠은 도전을 적습니다. 둘 다 맞습니다. 다르게 읽힌다는 사실을 확인하는 것이 목적입니다.

둘째, 학생이 게임 이름만 늘어놓고 끝내는 상황을 경계하세요. 활동이 잡담으로 흐르는 지점은 대개 M을 건너뛰고 A부터 말하기 시작할 때입니다. "먼저 규칙부터 적자"는 한마디로 되돌릴 수 있습니다.

셋째, 교사도 한 모둠을 맡아 함께 적어 보시기 바랍니다. 13쪽의 예시를 그대로 따라가면 5분이면 끝납니다.

> **진행 팁**
> "그 게임이 왜 재미있어요?"라는 발문에 학생들이 쏟아내는 대답이 곧 미학(A)의 목록입니다. 교사는 그 말을 받아 적기만 하면 됩니다.

1.9 헷갈리는 두 개념—GBL과 게이미피케이션

게임을 수업에 들이는 길은 크게 두 갈래입니다. Prensky가 정립한 게임기반학습(GBL)은 게임 그 자체를 학습 경험으로 삼는 접근이고, Deterding 등이 정의한 게이미피케이션은 게임이 아닌 활동에 게임의 요소만 이식하는 접근입니다. 목적과 설계 방법이 다르므로 구분해서 사용해야 합니다.[12]

두 말은 현장에서 자주 뒤섞여 쓰입니다. 포인트 제도를 운영하면서 게임기반학습을 한다고 말하거나, 학습 게임을 만들어 놓고 게이미피케이션이라고 부르는 경우가 흔합니다. 이름이 틀렸다고 수업이 나빠지지는 않지만, 무엇을 하고 있는지 모

[12] Prensky, M. (2001). *Digital game-based learning*. McGraw-Hill.; Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011). From game design elements to gamefulness: Defining "gamification". In *Proceedings of the 15th International Academic MindTrek Conference* (pp. 9–15). ACM.

르면 잘 안 될 때 어디를 고쳐야 할지도 알 수 없습니다.

게임기반학습(GBL) —"게임으로 배우기"

게임을 플레이하는 것 자체가 학습 경험이 되도록 설계합니다. 게임 속 문제 해결이 곧 학습 목표 도달입니다. 핵심은 내용이 게임 밖에 있지 않고 게임 안에 녹아 있다는 점입니다. 게임을 다 하고 나서 학습지를 푸는 것이 아니라, 게임을 하는 동안 학습이 일어납니다.

샌드박스 형태의 세계에서 지형을 만들며 지리를 배우거나, 분수가 적힌 물고기를 낚으며 분수 개념을 연습하는 활동이 여기에 해당합니다. 강점은 깊은 몰입과 맥락적 이해입니다. 유의점은 설계 난이도가 높다는 것, 그리고 학습 목표와 게임 목표의 정렬이 관건이라는 것입니다. 두 목표가 어긋나면 학생은 게임을 잘하기 위해 학습을 우회하는 방법을 찾아냅니다.

게이미피케이션—"게임처럼 배우기"

수업이나 과제처럼 게임이 아닌 활동에 게임의 요소만 가져와 동기를 높입니다. 포인트, 배지, 레벨, 리더보드, 도전과제가 대표적 장치입니다. 발표 포인트제, 과제 완료 배지, 모둠 순위판, 출석 스트리크가 교실에서 흔히 보는 형태입니다.

강점은 어떤 수업에도 즉시 적용할 수 있다는 것입니다. 개발도 필요 없고 준비물도 거의 없습니다. 유의점은 분명합니다. 보상이 목적이 되면 내적 동기를 해칠 수 있습니다. 원래 재미있어서 하던 일에 보상을 붙였다가 보상이 사라지자 아무도 하지 않게 되는 현상을 과잉정당화라고 부릅니다.[13]

[13] Lepper, M. R., Greene, D., & Nisbett, R. E. (1973). Undermining children's intrinsic interest with extrinsic reward. *Journal of Personality and Social Psychology, 28*(1), 129–137.

한눈에 비교하기

구분	게임기반학습(GBL)	게이미피케이션
한 줄 요약	게임으로 배우기	게임처럼 배우기
학습 내용의 자리	게임 안에 녹아 있음	게임 밖 활동에 그대로 있음
대표 장치	게임 속 문제 해결	포인트·배지·레벨·리더보드
강점	깊은 몰입과 맥락적 이해	어떤 수업에도 즉시 적용
유의점	설계 난이도, 목표 정렬	보상이 목적이 되면 내적 동기 저해

여기에 하나 더—게임리터러시는 어디에 있나

현장에서 더 자주 뒤섞이는 것은 사실 세 번째 말입니다. 게임기반학습과 게이미피케이션을 구분해 놓고도, 거기에 **게임리터러시**까지 나란히 올려놓고 셋 중 하나를 고르려 하면 계속 헷갈립니다. 셋이 같은 층에 있지 않기 때문입니다.

앞의 두 개념은 **수업을 어떻게 구성할 것인가**에 대한 답입니다. 곧 방법입니다. 게임리터러시는 **그래서 무엇을 할 줄 알게 되는가**에 대한 답입니다. 곧 능력입니다.[14] 방법과 능력은 고르는 관계가 아니라 짝을 짓는 관계입니다.

[14] Gee, J. P. (2003). *What video games have to teach us about learning and literacy.* Palgrave Macmillan.; Buckingham, D., & Burn, A. (2007). Game literacy in theory and practice. *Journal of Educational Multimedia and Hypermedia, 16*(3), 323–349.

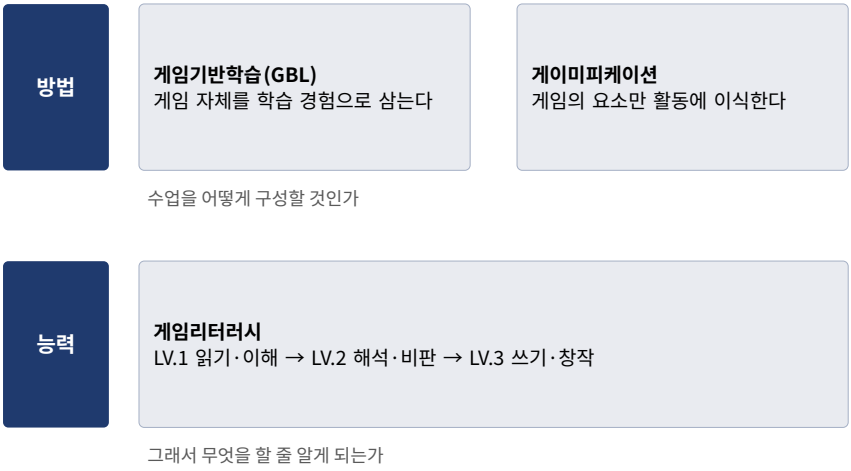


그림 1-3. 같은 층이 아니다. 앞의 둘은 방법이고, 게임리터러시는 그 방법으로 기르는 능력이다

층이 다르다는 말은 실무적으로 이런 뜻입니다. **게임기반학습으로 수업하면서 게임리터러시는 하나도 기르지 않을 수 있습니다.** 게임을 도구로만 쓰고 게임 자체를 한 번도 들여다보지 않으면 그렇게 됩니다. 반대로 **게임을 한 판도 만들지 않고 게임리터러시 수업을 할 수도 있습니다.** 학생들이 이미 하는 게임을 함께 분해해 보는 것만으로도 해석의 층위는 자랍니다.

구분	GBL · 게이미피케이션	게임리터러시
무엇에 대한 말인가	수업을 구성하는 방법	학습자가 갖추는 능력
묻는 질문	게임을 어떻게 수업에 들일까	게임을 읽고 만들 줄 아는가
게임의 자리	가르치기 위한 도구이자 장치	읽고 해석하고 쓰는 대상, 곧 텍스트
성공의 판단	학습 목표에 도달했는가	게임을 스스로 분석하고 설계할 수 있는가

자주 보는 네 가지 오해

1. **"게임으로 수업했으니 게임리터러시를 한 것이다."** 아닙니다. 분수 게임으로 분수를 가르쳤다면 학습 목표는 분수이지 게임이 아닙니다. 게임리터러시가

되려면 게임 자체를 들여다보는 활동이 한 번은 들어가야 합니다. "이 게임은 왜 계속하고 싶지?"라는 질문 하나면 시작됩니다.

2. **"포인트와 배지를 줬으니 게임리터러시다."** 게이미피케이션은 게임의 부품만 빌려온 것입니다. 학생은 게임을 읽은 적이 없습니다. 순위표를 운영하는 일과 순위표가 사람을 어떻게 움직이는지 아는 일은 다릅니다.
3. **"게임리터러시는 과몰입 예방 교육이다."** 절제와 자기조절은 LV.2 해석·비판의 한 갈래일 뿐입니다. 읽기(LV.1)와 창작(LV.3)이 빠지면 반쪽이 됩니다. 예방만 남으면 학생의 세계를 함께 읽는 일은 시작되지 않습니다.
4. **"게임리터러시는 게임 개발이나 코딩 교육이다."** 창작은 셋째 층위이지 전부가 아니고, 목적도 기술 습득이 아닙니다. 읽고 해석한 것을 자기 목적에 맞게 표현하는 행위입니다. 글쓰기 교육의 목표가 타자 연습이 아닌 것과 같습니다.

> **헷갈릴 때 던지는 질문 하나**
> "이 수업이 끝나고 학생에게 남는 것이 교과 지식인가, 게임을 보는 눈인가?" 앞이면 GBL이나 게이미피케이션을 쓴 교과 수업이고, 뒤면 게임리터러시 수업입니다. 둘 다 남는다면 가장 좋은 설계입니다.

이 책이 서 있는 자리도 이 구분으로 설명됩니다. 이 책은 먼저 **교사의 게임리터러시**를 LV.3 창작까지 끌어올리고(1–6단원), 그렇게 만든 게임으로 **GBL 수업**을 하게 하며, 거기에 **게이미피케이션 장치**를 하나 얹습니다. 셋이 나란히 놓이는 것이 아니라 순서대로 놓입니다.

이 교재의 전략은 결합입니다

둘 중 하나를 고를 필요는 없습니다. 이 교재는 교과 내용을 담은 게임(GBL)을 3–4단원에서 만들고, 여기에 실시간 리더보드라는 게이미피케이션 요소를 6단원에서 얹습니다. 게임 자체가 학습이 되게 하되, 다시 들어오게 만드는 장치를 바깥에 하나 두는 구조입니다.

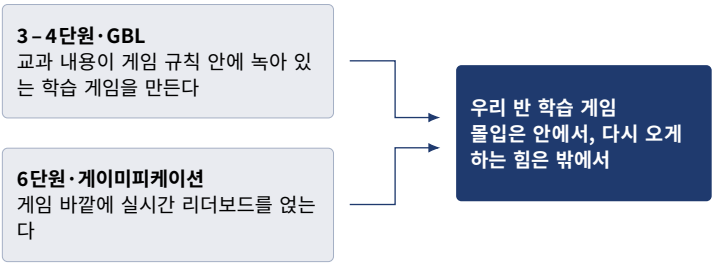


그림 1-4. 이 교재가 두 접근을 결합하는 방식

> **리더보드를 얹을 때의 원칙**
> 게이미피케이션 장치를 성적이나 상점에 연결하지 않는 것이 원칙입니다. 연결하는 순간 9쪽에서 본 마법의 원이 깨지고, 학생은 학습이 아니라 점수를 최적화하기 시작합니다.

1.10 교실 속 게임 활용—세 갈래 길

현재 학교 현장에서 게임을 활용하는 방식은 크게 세 갈래입니다. 각각 뚜렷한 장점과 한계가 있으며, 이 교재는 지금까지 진입 장벽이 가장 높았던 세 번째 길을 엽니다. 세 갈래는 우열이 아니라 용도가 다르므로, 셋 다 알아 두고 상황에 맞게 고르는 편이 좋습니다.

첫째, 기성 교육용 게임

교육용으로 개발된 완성형 게임을 그대로 수업에 활용하는 방식입니다. 샌드박스형 플랫폼이나 교육용 게임 플랫폼이 여기에 속합니다. 세계 재현, 협력 건축, 코딩 학습 등 완성도 높은 콘텐츠를 즉시 쓸 수 있고 관련 수업 사례도 풍부합니다.

장점은 높은 품질과 검증된 사례입니다. 다른 교사가 이미 해 본 수업이 있으니 지도안을 구하기도 쉽습니다. 한계는 우리 반 진도나 수준에 맞춘 세부 조정이 불가능하다는 점입니다. 이번 주에 배운 어휘를 게임 안에 넣을 수는 없습니다.

둘째, 퀴즈 게임 플랫폼

문항만 입력하면 즉시 경쟁형 복습 게임이 되는 플랫폼입니다. 실시간 퀴즈나 골든벨형 도구가 여기에 해당합니다. 준비 부담이 적어 전 과목에서 가장 널리 쓰이며, 실시간 순위 공개로 수업 열기를 끌어올립니다.

장점은 간편함과 즉시성입니다. 10분이면 한 차시 분량의 복습 게임이 만들어집니다. 한계는 게임 형식이 퀴즈로 고정되어 확장이 어렵다는 것입니다. 문항을 바꿀 수는 있어도 규칙을 바꿀 수는 없으므로, 매번 같은 경험이 반복됩니다.

셋째, 교사 자작 웹게임

내용과 규칙, 난이도와 디자인을 모두 교사가 결정하는 맞춤형 게임입니다. 지금까지는 코딩 역량이 필요해 소수의 길이었지만, AI 에이전트의 등장으로 모든 교사에게 열렸습니다. 이 교재가 여는 길이 여기입니다.

방식	장점	한계
기성 교육용 게임	높은 품질과 검증된 사례	우리 반에 맞춘 세부 조정 불가
퀴즈 게임 플랫폼	간편함과 즉시성	형식이 퀴즈로 고정, 확장 어려움
교사 자작 웹게임	내용·규칙·난이도를 모두 결정	지금까지는 코딩이 진입 장벽

세 번째 길을 연다고 해서 앞의 두 길을 버리라는 뜻은 아닙니다. 학기 초 복습은 퀴즈 플랫폼으로, 한 단원을 통째로 다루는 프로젝트는 기성 게임으로, 이번 주 학습 목표에 딱 맞는 5분짜리 연습은 자작 게임으로 나누는 편이 현실적입니다.

1.11 비판적으로 읽기—게임이 나를 붙잡는 장치들

상업 게임은 이용자의 시간과 지갑을 붙잡도록 정교하게 설계됩니다. 게임리터러시의 "해석·비판" 층위는 바로 이 설계를 읽어내는 능력입니다. 대표적인 세 가지 장치를 해부해 봅니다.

먼저 전제를 하나 분명히 해 둡니다. 이 절의 목적은 게임을 나쁜 것으로 규정하는 데 있지 않습니다. 설계에는 의도가 있고, 그 의도를 읽어낼 수 있는 사람은 휘둘리지 않는다는 것이 요점입니다. 학생에게 가르쳐야 할 것도 게임에 대한 혐오가 아니라 설계도를 읽는 눈입니다.

보상 루프

언제 나올지 모르는 보상, 곧 가변 비율 보상이 행동을 가장 강하게 붙잡습니다. 확실하게 열 번에 한 번 나오는 보상보다, 언제 나올지 모르는 보상이 훨씬 오래 사람을 붙잡아 둡니다. 슬롯머신과 동일한 심리 기제이며, "한 판만 더"를 만들어내는 핵심 장치입니다.[15]

이 장치는 행동주의 심리학의 강화 계획 개념으로 설명할 수 있어 교과 연계도 가능합니다. 실제로 학생에게 "왜 그만두기가 어려운지"를 설명할 때 이보다 설득력 있는 언어가 없습니다. 자신의 의지가 약해서가 아니라 그렇게 되도록 설계되었다는 사실을 아는 것만으로 학생의 자책이 줄어듭니다.

확률형 아이템

낮은 확률의 희귀 아이템을 반복 구매로 유도하는 수익 모델입니다. 표시 확률로 기댓값을 계산해 보면 구조가 선명히 보입니다. 0.5%짜리 아이템을 얻기까지 평균 몇 번을 시도해야 하는지, 그 시도에 얼마가 드는지를 직접 계산해 보는 순간 학생의 표정이 달라집니다.[16]

실제로 확률 계산은 초등 고학년 – 중등 수학 수업의 훌륭한 실생활 소재가 됩니다.

[15]가변 비율 강화 계획의 원 논의는 다음과 같습니다. Ferster, C. B., & Skinner, B. F. (1957). *Schedules of reinforcement*. Appleton-Century-Crofts.

[16]확률형 아이템 이용과 문제적 도박 성향의 관련을 다룬 대규모 조사로 다음이 자주 인용됩니다. Zendle, D., & Cairns, P. (2018). Video game loot boxes are linked to problem gambling: Results of a large-scale survey. *PLOS ONE, 13*(11), e0206767. 본문의 0.5%는 계산 방법을 보이기 위한 예시 수치이며 특정 게임의 값이 아닙니다.

교과서의 주머니 속 구슬보다 학생이 어제 뽑은 캐릭터가 더 좋은 자료입니다. 다만 특정 게임을 지목해 비난하는 자리가 되지 않도록, 확률 표기 자체를 자료로 삼는 편이 안전합니다.

다크 패턴

출석 보상, 한정 이벤트, 랭킹 압박, 소셜 알림처럼 이탈을 막기 위해 설계된 장치의 총칭입니다. Zagal 등의 연구는 게임 속 다크 패턴을 시간·금전·사회관계 착취로 유형화했습니다.[17] 하루라도 접속하지 않으면 연속 보상이 끊기는 구조는 시간을, 지금 사지 않으면 사라지는 한정 상품은 금전을, "네가 안 들어와서 우리 팀이 졌다"는 압박은 사회관계를 재료로 씁니다.

장치의 이름을 아는 것만으로도 영향력이 줄어듭니다. 이름이 붙지 않은 압박은 그냥 불편한 감정으로 남지만, "이건 이탈 방지 장치구나"라고 부를 수 있게 되면 거리를 둘 수 있습니다.

> **교육의 방향—금지가 아니라 해부**
>
> 구조를 아는 학생은 휘둘리지 않습니다. 학생과 함께 게임의 설계를 분석하는 수업은 그 자체로 실질적인 과몰입 예방교육이 됩니다. 게임을 못 하게 막는 수업보다, 왜 그만두기 어려운지를 함께 뜯어보는 수업이 오래 남습니다.

수업으로 옮길 때

이 세 장치는 그대로 한 차시가 됩니다. 16쪽의 MDA 해부 활동을 한 번 해 본 학급이라면, 같은 형식으로 "이 게임이 나를 붙잡는 규칙 찾기"를 진행할 수 있습니다. 학생이 찾아낸 장치에 이름을 붙여 주는 것이 교사의 역할입니다.

[17] Zagal, J. P., Björk, S., & Lewis, C. (2013). Dark patterns in the design of games. In *Proceedings of the 8th International Conference on the Foundations of Digital Games.*

1.12 소비자에서 생산자로—왜 교사가 만들어야 하는가

기성 게임을 고르는 것과 직접 만드는 것 사이에는 질적인 차이가 있습니다. 게임 리터러시의 관점에서, 그리고 교실의 현실에서, 교사가 생산자가 되어야 할 이유는 분명합니다. 이유는 세 가지입니다.

첫째, 창작이 가장 깊은 리터러시입니다

직접 만들어 본 사람만이 게임의 설계 의도를 꿰뚫어 봅니다. 글쓰기가 독해력을 완성하듯, 게임 제작 경험은 게임을 읽는 눈을 완성합니다. 한 번이라도 난이도 곡선을 조정해 본 사람은 다른 게임의 3단계가 왜 그 자리에 있는지를 다르게 봅니다.

7쪽의 계단으로 말하면, LV.3(창작)에 오른 교사는 LV.2(비판)의 깊이가 달라집니다. 층위는 위로만 올라가는 것이 아니라 아래를 되비춥니다.

둘째, 우리 반 맞춤 게임은 시중에 없습니다

우리 반의 진도, 우리 학생들의 수준, 이번 주에 배운 어휘가 담긴 게임은 어느 스토어에도 없습니다. 있을 수가 없습니다. 그 조합을 아는 사람이 세상에 한 명뿐이기 때문입니다. 교육과정과 학급 맥락을 아는 유일한 사람, 즉 교사만이 만들 수 있습니다.

셋째, 만드는 교사가 학생도 만들게 합니다

교사의 창작 경험은 곧 학생 게임 제작 수업의 설계 역량이 됩니다. 제작 과정의 막힘과 해법을 몸으로 아는 교사는 학생의 창작 프로젝트를 지도할 수 있는 코치가 됩니다. 반대로 한 번도 만들어 본 적 없는 교사는 학생이 막혔을 때 무슨 말을 해 주어야 할지 알 수 없습니다.

그동안의 장벽, 그리고 남은 질문

지금까지 이 세 가지 이유를 다 알고도 시작하지 못한 이유는 하나였습니다. 코딩입니다. 아이디어를 화면 위의 무언가로 바꾸려면 문법을 배워야 했고, 대부분의 교사에게 그 비용은 감당할 만한 것이 아니었습니다.

그 장벽이 사라졌습니다. 아이디어를 한국어로 설명하면 앱이 되어 돌아오는 시대이고, 남은 질문은 "무엇을 만들 것인가" 뿐입니다. 그리고 그 질문에 답하는 데 필요한 것이 바로 이 단원에서 익힌 언어입니다. 무엇을 만들지 정하려면 재미의 이름을 알아야 하고, 규칙과 재미의 연결 고리를 말할 수 있어야 합니다.

연수에서 자주 나오는 질문 셋

Q. 게임을 수업에 들이면 학생이 더 게임에 빠지지 않을까요?

A. 금지가 아니라 해부가 이 교재의 방향입니다(24쪽). 구조를 아는 학생은 휘둘리지 않으며, 설계를 함께 분석하는 수업은 그 자체로 실질적인 과몰입 예방교육이 됩니다.

Q. GBL과 게이미피케이션 중 무엇부터 시작해야 하나요?

A. 둘 중 하나를 고르는 문제가 아닙니다(17쪽). 이 교재는 교과 내용을 담은 게임을 3–4단원에서 만들고, 리더보드를 6단원에서 얹어 두 접근의 강점을 함께 취합니다.

Q. 코딩을 전혀 모르는데 창작 단계까지 갈 수 있나요?

A. 갈 수 있습니다. 코딩은 AI 에이전트가 대신 씁니다. 다만 무엇을 만들지는 교사가 정해야 하고, 그 언어가 이 단원의 내용입니다. 도구 이야기는 2단원에서 시작합니다.

1.13 함께 가는 길—게임리터러시 지원 생태계

게임리터러시 교육은 개별 교사의 열정만으로 굴러가지 않습니다. 국내에는 이미 공적 기관과 교사 공동체로 이루어진 지원 생태계가 갖추어져 있으며, 연수 이후의 성장은 이 생태계에 접속할 때 지속됩니다.

혼자 만들면 두 번째 게임에서 멈춥니다. 첫 게임은 의욕으로 만들어지지만, 그다음부터는 막히는 지점이 반복되고 물어볼 곳이 없으면 그대로 끝납니다. 아래 세 갈래는 그 지점에서 붙잡을 수 있는 손잡이입니다.

게임문화재단과 한국콘텐츠진흥원—공적 기반

전국 단위의 게임리터러시 교육 사업과 게임과몰입 상담 인프라를 운영하고, 게임이용자 실태조사 등 정책 연구의 기반 데이터를 매년 발간합니다. 교사가 수업 근거로 삼을 수 있는 공신력 있는 통계와 자료의 출처입니다.[18] 3쪽에서 인용한 이용률 숫자도 이 조사에서 나온 것입니다.

학부모 연수나 학교 계획서에 근거가 필요할 때, 개인 블로그의 숫자 대신 여기서 발간한 자료를 쓰시기 바랍니다. 매년 갱신되므로 최신 연도판을 확인하는 습관도 함께 들이면 좋습니다.

게임문화교육연수원—공식 연수 기관

2026년 개원한 교원·보호자 대상 공식 연수 기관입니다. 게임문화 이해와 게임리터러시 교육과정을 다루는 공식 원격 연수를 제공하며, 개원 이후 교원 대상 신규 과정을 지속 개설하고 있습니다.[19] 이 교재를 이수한 뒤의 심화 학습 경로로 적합합니다.

게임리터러시 연구교사 프로그램—실천 공동체

전국 초·중등 교사들이 게임 활용 수업을 연구하고 사례를 공유하는 실천 공동체입니다. 혼자 만들다 막히는 지점을 함께 넘고, 서로의 산출물을 검증해 주는 동료 네트워크가 지속 성장의 열쇠입니다. 만든 게임을 한 번이라도 다른 교사에게 보여 준 경험이 있으면 다음 게임의 완성도가 달라집니다.

> **연수 운영 팁**
> 소속 시도교육청의 게임리터러시 관련 연수·연구회 정보를 마지막 차시에 함께 안내하면 연수의 효과가 현장으로 이어집니다. 연수가 끝나는 날 다음 접속 지점을 손에 쥐여 주는 것이 핵심입니다.

[18] 한국콘텐츠진흥원. (2025). *2025 게임이용자 실태조사*. 한국콘텐츠진흥원.

[19] 게임문화재단. (2026). *게임문화교육연수원*. https://www.game-edu.kr 개설 과정은 해마다 달라지므로 최신 공고를 확인하십시오.

1.14 단원 마무리

여기까지가 이론편입니다. 손으로 만든 것은 없지만, 다음 단원부터 쓸 도구를 손에 넣었습니다.

이 단원에서 우리는 먼저 숫자에서 출발했습니다. 전 국민 게임 이용률 50.2%, 10대 80.9%. 학생에게 게임은 여가가 아니라 생활 공간이라는 사실을 확인하고, 그 세계를 읽는 능력에 게임리터러시라는 이름을 붙였습니다. 리터러시가 문자에서 미디어와 디지털을 거쳐 게임으로 넓어져 온 경로를 따라가며, 게임이 규칙과 서사와 상호작용이 결합된 가장 복합적인 텍스트임을 확인했습니다.

이어서 세 층위의 계단을 놓았습니다. 온전히 플레이하는 읽기, 왜 재미있는지 말로 설명하는 해석, 학습 목표를 규칙으로 번역하는 창작. 그 계단의 이론적 뿌리로 호이징가의 마법의 원과 카이와의 네 가지 놀이 분류를 살폈고, 게임을 세 겹으로 분해하는 MDA를 익혀 달리기 점프 게임에 직접 적용해 보았습니다. 여덟 가지 미학의 이름을 손에 넣었으니 이제 "재미있게"라는 말 대신 "도전과 발견을"이라고 말할 수 있습니다.

후반부는 판단의 언어였습니다. 게임으로 배우기와 게임처럼 배우기를 구분하고, 교실에서 게임을 쓰는 세 갈래 길 중 이 교재가 여는 세 번째 길을 확인했습니다. 보상 루프와 확률형 아이템과 다크 패턴을 해부하며 금지가 아닌 해부의 관점을 세웠고, 마지막으로 교사가 소비자를 넘어 생산자가 되어야 하는 세 가지 이유를 정리했습니다.

획득한 아이템—네 줄 요약

핵심 개념	한 줄 요약
게임리터러시 3층위	게임을 읽고(플레이) 해석하고(비판) 쓰는(창작) 능력. 창작이 리터러시를 완성하며, 이 교재의 목표는 LV.3입니다
MDA 분석 렌즈	규칙(M)이 상호작용(D)을 낳고 재미(A)로 완성되는 구조. 분석할 땐 A에서 M으로, 설계할 땐 M에서 A로 오갑니다
GBL·게이미피케이션	게임으로 배우기와 게임처럼 배우기. 이 교재는 GBL 게임 제작에 리더보드를 결합합니다
비판적 읽기 방패	보상 루프·확률형 아이템·다크 패턴. 금지 대신 해부, 구조를 아는 학생은 휘둘리지 않습니다

남은 숙제. 학생들이 요즘 가장 많이 하는 게임 하나를 골라 13쪽의 표대로 세 칸을 채워 보세요. 그리고 "이 게임의 (A)는 (M)에서 나온다" 문장을 한 줄 완성해 적어 두시기 바랍니다. 그 한 줄이 3단원에서 만들 게임의 첫 기획서가 됩니다.

다음 단원 예고. WORLD 2 · AI 에이전트와 게임 개발의 만남—코딩이라는 마지막 장벽이 어떻게 사라졌는지, 그리고 교사가 한국어로 설명한 아이디어가 어떻게 실행되는 앱이 되는지를 다룹니다.

게임을 읽을 수 있는 교사는 학생의 세계를 절반 더 읽습니다.

참고문헌

Buckingham, D. (2003). *Media education: Literacy, learning and contemporary culture.* Polity Press.

Buckingham, D., & Burn, A. (2007). Game literacy in theory and practice. *Journal of Educational Multimedia and Hypermedia, 16*(3), 323–349.

Caillois, R. (2001). *Man, play and games.* University of Illinois Press. (원저 1958년 출판)

Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011). *From game design elements to gamefulness: Defining gamification.* MindTrek 2011.

Ferster, C. B., & Skinner, B. F. (1957). *Schedules of reinforcement.* Appleton-

Century-Crofts.

Gee, J. P. (2003). *What video games have to teach us about learning and literacy.* Palgrave Macmillan.

Gilster, P. (1997). *Digital literacy.* Wiley.

Hunicke, R., LeBlanc, M., & Zubek, R. (2004). *MDA: A formal approach to game design and game research.* AAAI Workshop.

Huizinga, J. (1955). *Homo ludens.* Beacon Press. (원저 1938년 출판)

Lepper, M. R., Greene, D., & Nisbett, R. E. (1973). Undermining children's intrinsic interest with extrinsic reward. *Journal of Personality and Social Psychology, 28*(1), 129–137.

New London Group. (1996). A pedagogy of multiliteracies: Designing social futures. *Harvard Educational Review, 66*(1), 60–92.

Prensky, M. (2001). *Digital game-based learning.* McGraw-Hill.

Salen, K., & Zimmerman, E. (2004). *Rules of play: Game design fundamentals.* MIT Press.

Zagal, J. P., Björk, S., & Lewis, C. (2013). *Dark patterns in the design of games.* FDG 2013.

Zendle, D., & Cairns, P. (2018). Video game loot boxes are linked to problem gambling: Results of a large-scale survey. *PLOS ONE, 13*(11), e0206767.

게임문화재단. (2026). *게임문화교육연수원*. https://www.game-edu.kr

한국콘텐츠진흥원. (2025). *2025 게임이용자 실태조사*.

WORLD 2

AI 에이전트와 게임 개발

대화가 곧 개발이 되는 자리

1단원에서 좋은 학습 게임이 무엇인지 살펴보았다면, 이 단원은 그것을 실제로 만들 수 있게 해 주는 기술의 지형을 다룹니다. 챗봇과 구분되는 AI 에이전트의 작동 원리, 자연어 대화로 소프트웨어를 만드는 바이브 코딩, 그리고 이 책의 실습 도구인 Google AI Studio까지 차례로 정리합니다.

단원 끝의 준비 점검을 마치면, 3단원부터는 곧바로 첫 게임 제작에 들어갑니다.

이 단원은 이 책에서 유일하게 화면 캡처가 없는 단원입니다. 손을 움직이기 전에 머리로 정리해 둘 것을 다루기 때문입니다. 다만 여기서 다루는 개념은 모두 3단원 이후의 실습 장면과 하나씩 짝지어 두었습니다. "에이전트가 스스로 반복한다"는 문장은 3단원에서 생성 화면을 지켜보는 1–2분으로, "교사가 교육 설계를 맡는다"는 문장은 4단원에서 문항을 손보는 시간으로 돌아옵니다.

이 단원을 마치면 다음 네 가지를 할 수 있어야 합니다. 첫째, 챗봇과 AI 에이전트의 차이를 작동 방식으로 설명할 수 있습니다. 둘째, 교육 현장에서 에이전트가 이미 쓰이고 있는 영역을 파악합니다. 셋째, 바이브 코딩의 원리와 그로 인한 교사 역할의 변화를 이해합니다. 넷째, Google AI Studio의 핵심 기능을 조망하고 계정과 브라우저 점검까지 마칩니다. 마지막 항목은 실습 환경 점검이므로 3단원을 시작하기 전에 반드시 끝내 두어야 합니다.

> **이 단원을 여는 한 문장**
>
> "이제 가장 인기 있는 프로그래밍 언어는 영어다."
>
> —안드레 카파시(전 테슬라 AI 총괄), 바이브 코딩 개념의 명명자[1]

2.1 AI 에이전트란—챗봇과 무엇이 다른가

인공지능 분야에서 에이전트(agent)란 환경을 인식하고 목표 달성을 위해 스스로 행동하는 주체를 뜻합니다(Russell & Norvig, 2021).[2] 최근의 AI 에이전트는 여기에 대규모 언어모델이 결합되면서 성격이 달라졌습니다. 도구 사용과 계획, 기억이 함께 붙으면서 질문에 답하는 수준을 넘어 여러 단계의 일을 끝까지 수행하게 된 것입니다.

교사에게 익숙한 쪽은 챗봇입니다. 한 번 묻고 한 번 답하는 문답 구조이고, 작업의 모든 단계를 사람이 지시해야 합니다. 답변이 아무리 좋아도 그것을 한글 문서로 옮기고 학습지 모양으로 다듬는 일은 여전히 사람의 손에 남습니다. 산출물이 텍스트 답변이기 때문입니다. 아이디어를 얻거나 문장을 다듬거나 개념을 물어볼 때

[1] 인용문과 "바이브 코딩"은 각각 다른 시점의 게시물입니다. Karpathy, A. (2023). "The hottest new programming language is English" [게시물]. X.; Karpathy, A. (2025). "Vibe coding" [게시물]. X.

[2] Russell, S., & Norvig, P. (2021). *Artificial intelligence: A modern approach* (4th ed.). Pearson.

는 이 방식이 가장 빠릅니다.

AI 에이전트는 목표를 주면 스스로 계획을 세우고 단계를 나눕니다. 코드 실행, 파일 생성, 검색 같은 도구를 직접 사용하며 여러 단계를 연속으로 수행하고, 도중에 오류가 나면 스스로 고쳐 다시 시도합니다. 산출물이 텍스트가 아니라 완성된 결과물—문서, 앱, 게임—이라는 점이 결정적인 차이입니다. "퀴즈 게임 만들어줘" 한마디로 실제로 동작하는 게임이 나옵니다.

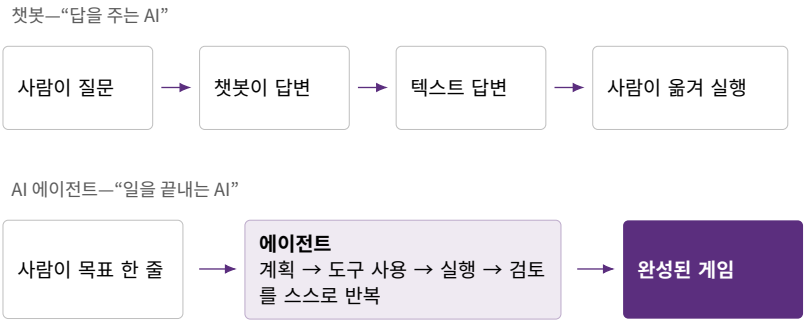


그림 2-1. 챗봇과 AI 에이전트. 사람이 어디까지 손을 대야 하는지가 다르다

구분	챗봇	AI 에이전트
한 문장	답을 주는 AI	일을 끝내는 AI
작동 방식	한 번 묻고 한 번 답하는 문답	계획을 세우고 여러 단계를 연속 수행
사람의 몫	모든 단계를 지시하고 직접 실행	목표 제시와 중간 방향 조정
도구 사용	없음. 답변 안에서 끝남	코드 실행·파일 생성·검색을 직접 호출
오류 처리	사람이 알아채고 다시 질문	스스로 찾아 수정하고 재시도
산출물	텍스트 답변	완성된 문서·앱·게임
활용 예	아이디어 얻기, 문장 다듬기	동작하는 학습 게임 완성

이 차이가 교사에게 갖는 의미는 분명합니다. 챗봇 시대에는 AI의 답을 수업 자료로 "옮기는" 일이 늘 남아 있었습니다. 에이전트 시대에는 완성물이 바로 나옵니다. 게임 제작처럼 여러 단계가 필요한 일이 한 번의 대화로 가능해진 것이고, 이

책의 실습이 성립하는 근거도 여기에 있습니다.

2.2 에이전트는 어떻게 일하는가—다섯 단계 루프

에이전트의 작동은 다섯 단계의 루프로 이해할 수 있습니다.[3] 사람이 하는 일은 처음의 목표 제시와 중간중간의 방향 조정뿐이며, 나머지는 에이전트가 만족스러운 결과가 나올 때까지 스스로 반복합니다. 3단원에서 생성 화면을 지켜볼 때 지금 어느 단계인지 짚어 보면 이해가 빨라집니다.

1. **목표 접수.** "분수 낚시 게임 만들어줘"처럼 자연어로 된 목표를 받아 요구사항을 해석합니다. 사람이 개입하는 지점은 사실상 여기 하나입니다.
2. **계획 수립.** 화면 구성, 게임 규칙, 문항 구조 등 해야 할 일의 순서를 스스로 설계합니다. 요청에 없던 항목도 이 단계에서 채워집니다.
3. **도구 사용.** 코드 작성, 파일 생성, 이미지 준비 등 필요한 도구를 직접 호출합니다.
4. **실행·생성.** 게임을 실제로 만들어 미리보기로 실행하고 결과를 확인합니다.
5. **검토·수정.** 오류를 찾아 고치고, 처음 요청과 결과를 비교해 부족한 부분을 보완합니다.

[3] 다섯 단계 구분은 Russell, S., & Norvig, P. (2021). *Artificial intelligence: A modern approach* (4th ed.). Pearson. 의 에이전트 개념(인식 → 판단 → 행동)을 실습 관찰에 맞게 이 책이 다시 나눈 것입니다. 서비스마다 내부 동작과 표현은 다를 수 있습니다.

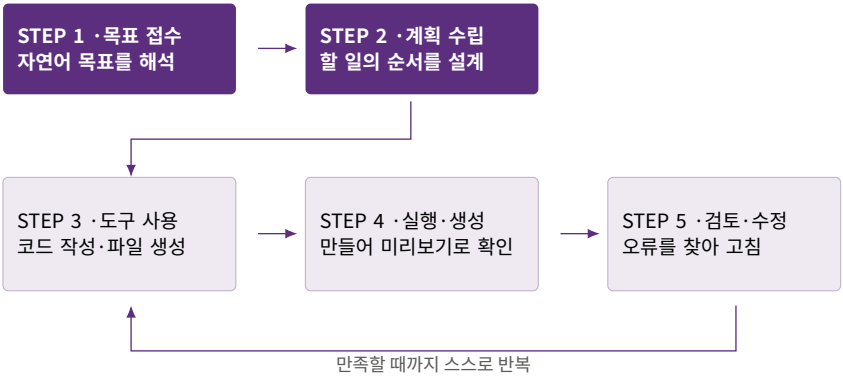


그림 2-2. 에이전트의 다섯 단계 루프. 사람이 하는 일은 첫 단계와 방향 조정뿐이다

핵심은 마지막 세 단계가 한 번으로 끝나지 않는다는 점입니다. 에이전트는 도구 사용부터 검토·수정까지를 만족할 때까지 스스로 반복합니다. 교사는 그 과정을 지켜보다가 결과를 보고 대화로 방향만 잡아 주면 됩니다. 생성에 1–2분이 걸리는 것도 이 반복 때문입니다.

> **에이전트 시대의 교사 역량**
> 여기서 필요한 역량은 코딩 실력이 아닙니다. 첫째, 좋은 목표를 명확히 제시하는 능력, 즉 프롬프트입니다. 둘째, 나온 결과물을 교육적으로 검토하는 안목입니다. 이 두 가지가 이 책 전체를 관통하는 훈련 목표입니다.

2.3 교육 현장의 AI 에이전트—이미 시작된 변화

AI 에이전트는 이미 학교 현장 곳곳에서 쓰이고 있습니다. 시도교육청 단위의 교사 연수도 "AI 활용"을 넘어 "AI 에이전트 직접 제작"으로 진화하는 중입니다. 대표적인 네 가지 활용 영역을 살펴봅니다.

수업 자료 제작

학습지, 평가 문항, 프레젠테이션을 교육과정 성취기준에 맞춰 통째로 생성합니다. 중요한 것은 초안 생성에서 끝나지 않는다는 점입니다. "2번 문항이 너무 어려우니 낮춰 줘"처럼 수정과 보완까지 대화로 반복할 수 있어, 자료 제작에 드는 시간이 크게 줄어듭니다.

학교 업무 자동화

가정통신문 초안 작성, 명렬표 정리, 일정 관리 같은 반복적인 행정 업무를 에이전트에게 위임합니다. 절약된 시간은 수업 연구와 학생 상담이라는 교사 본연의 일로 돌아갑니다. 다만 학생 정보가 들어가는 업무는 47쪽의 개인정보 원칙을 먼저 확인해야 합니다.

교사 연수의 확산

여러 시도교육청이 교사가 직접 수업용 AI 에이전트를 설계·제작하는 직무연수를 개설하고 있습니다.[4] "쓰는 교사"에서 "만드는 교사"로의 전환이 개인의 취향이 아니라 정책 차원에서 진행 중이라는 뜻입니다.

학습 게임 제작

교과 내용을 담은 웹게임을 교사가 직접 개발합니다. 이 책의 핵심 활용처이고, 3단원부터 Google AI Studio로 이 과정을 단계별로 실습합니다.

네 영역의 공통점은 하나입니다. "AI에게 물어보는 교사"에서 "AI에게 일을 맡기는 교사"로의 이동입니다. 다음 절에서 그 방법론인 바이브 코딩을 다룹니다.

[4]연수 개설 현황은 시도와 연도에 따라 크게 다릅니다. 정확한 과정명과 일정은 소속 교육청의 직무연수 공고에서 확인하십시오.

2.4 바이브 코딩—대화가 곧 개발이다

2025년 초 안드레 카파시가 이름 붙인 "바이브 코딩(vibe coding)"은 개발의 문법을 바꾸었습니다.[5] 원하는 것을 자연어로 설명하면 AI가 코드를 쓰고, 사람은 결과를 보며 대화로 다듬는 방식입니다. 전문 개발자에게는 생산성 도구가 되었고, 비개발자에게는 소프트웨어 제작에 진입하는 문이 되었습니다.

정의는 짧습니다. 바이브 코딩이란 코드를 직접 작성하지 않고, 자연어 대화로 AI에게 의도를 설명하며 소프트웨어를 만들어 가는 개발 방식입니다. 여기서 바뀐 것은 도구가 아니라 시작점입니다. 예전에는 "무엇을 만들 수 있는가"가 배운 문법의 범위 안에서 결정되었지만, 이제는 "무엇을 만들고 싶은가"를 설명할 수 있으면 됩니다.

구분	전통적 개발	바이브 코딩
시작 조건	프로그래밍 언어 문법을 먼저 배워야 시작	하고 싶은 것을 한국어로 설명하면 시작
코드 작성	한 줄 한 줄 사람이 직접 작성	AI가 작성·실행·수정을 담당
수정 방식	오류 해결에 몇 시간씩 소요	결과를 보며 "여기를 이렇게 바꿔줘"
소요 기간	아이디어에서 완성까지 수 개월	아이디어에서 동작하는 게임까지 수십 분
교사에게	사실상 진입 불가의 영역	교육 전문성만 있으면 누구나 제작

> **검증의 책임은 사람에게 남습니다**
>
> 바이브 코딩에서도 결과를 검증할 책임은 사람에게 있습니다. 교육 자료라면 더욱 그렇습니다. AI가 만든 문항에 사실 오류나 편향이 섞여 있을 수 있고, 그것을 걸러내는 것은 교사의 일입니다. 47쪽의 안전 수칙에서 다시 다룹니다.

[5] Karpathy, A. (2025). "Vibe coding" [게시물]. X.

2.5 역할 분담—교사의 전문성이 곧 개발력

바이브 코딩은 "누가 무엇을 잘해야 하는가"를 다시 정의합니다. 기술 구현은 AI의 몫이 되었고, 남은 것은 교육의 몫입니다. 무엇을 가르칠지, 어떤 재미로 배우게 할지, 결과물이 교육적으로 타당한지의 판단. 1단원의 이론이 여기서 기획 언어로 쓰입니다.

AI 에이전트가 맡는 것

- 프로그래밍 문법과 코드 작성 전체
- 화면 디자인과 상호작용 구현
- 오류(버그) 찾기와 수정
- 배포·저장 등 기술적 처리(GitHub, 클라우드)

이 네 가지는 교사가 손대지 않아도 되는 영역입니다. 코드를 읽을 줄 몰라도 실습에 지장이 없다는 말은 과장이 아니라 여기서 나옵니다.

교사가 맡는 것

- 학습 목표와 성취기준을 게임 내용으로 번역하기
- 학생 수준에 맞는 난이도와 문항 설계
- 게임 규칙과 재미의 기획—1단원의 MDA가 바로 이 언어입니다
- 결과물의 교육적 검토와 품질 최종 판단

이 네 가지는 AI가 대신할 수 없습니다. 우리 반 학생이 어디서 막히는지, 어떤 문항이 오개념을 드러내는지는 교실에 있는 사람만 압니다.

그래서 1단원의 게임리터러시와 MDA가 먼저였습니다. 기술 장벽이 사라진 시대에 좋은 학습 게임과 그렇지 않은 게임을 가르는 것은 코딩 실력이 아니라 교육 설계의 질입니다. 기술은 AI가, 교육은 교사가. 이것이 새로운 분업입니다.

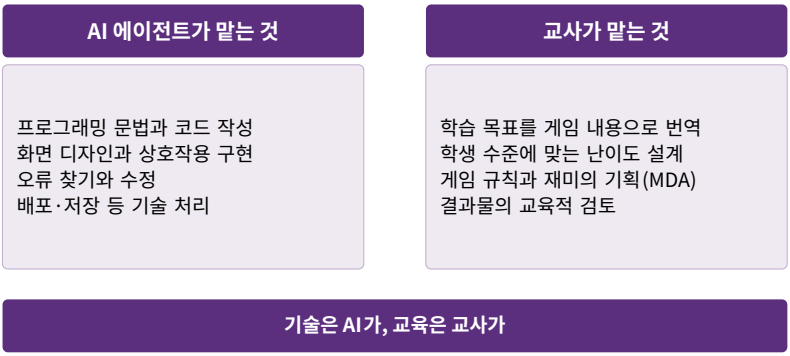


그림 2-3. 새로운 분업. 기술 장벽이 사라진 자리에 교육 설계가 남는다

2.6 바이브 코딩 도구 세 가지

바이브 코딩을 지원하는 대표 도구는 세 가지입니다. 셋 다 자연어로 앱을 만들 수 있어 어느 것을 골라도 게임은 나옵니다. 다만 게임 제작 연수의 관점에서는 기준이 하나 더 붙습니다. 생성에서 배포를 거쳐 백엔드(리더보드)까지 한곳에서 끝나는가입니다.

Google AI Studio —이 책의 선택

Gemini를 기반으로 하며, 웹 브라우저에서 열리는 빌드(Build) 모드에서 작업합니다. 강점은 생성에서 배포, 백엔드까지가 한 곳에서 이어진다는 점입니다. 배포는 공유 링크, GitHub, Cloud Run 세 갈래로 열려 있고, 리더보드에 필요한 Firebase가 자동으로 연동됩니다. 비용은 무료이며 구글 계정만 있으면 됩니다.

Claude —아티팩트(Artifacts)

대화 중에 미리보기가 함께 생성되는 방식입니다. 긴 맥락을 다루는 능력과 코드 품질이 강점으로 꼽힙니다. 배포는 아티팩트 링크 공유로 이루어지고, 백엔드가 필요하면 별도 서비스를 연결해야 합니다. 무료 사용이 가능하고 유료 플랜이 함께 있습니다.

ChatGPT —캔버스(Canvas)

캔버스 화면에서 코드를 편집하는 방식입니다. 대중성과 범용성이 강점입니다. 다만 배포하려면 코드를 복사해 별도 호스팅에 올려야 하고, 백엔드도 별도 서비스를 연결해야 합니다. 무료 사용이 가능하고 유료 플랜이 함께 있습니다.

셋 다 훌륭한 도구이며 수업 목적에 따라 선택이 달라질 수 있습니다.[6] 이 책은 무료로 제작부터 배포와 리더보드까지 한곳에서 완결되는 Google AI Studio를 실습 도구로 채택합니다. 연수에서 참가자 전원이 같은 화면을 보며 같은 속도로 움직여야 한다는 점도 함께 고려한 결정입니다.

2.7 왜 Google AI Studio인가—네 가지 결정타

교사 연수 도구는 성능만으로 선택할 수 없습니다. 진입 장벽(비용과 가입 절차), 결과물의 수업 적용 가능성(배포), 확장성(백엔드)까지 갖춰야 합니다. AI Studio는 네 조건을 모두 충족합니다.

[6] 세 도구의 기능·배포 방식·요금은 각 서비스의 공식 안내 문서를 기준으로 2026년 8월에 정리한 것입니다. 인공지능 서비스의 기능과 요금 정책은 수시로 바뀌므로, 연수 직전에 각 서비스의 최신 안내를 다시 확인하십시오. Google AI Studio에 관한 서술의 근거는 다음과 같습니다. Google. (2026). *Build apps in Google AI Studio.* Gemini API 개발자 문서. https://ai.google.dev/gemini-api/docs/aistudio-build-mode

1. **완전 무료 시작.** 구글 계정만 있으면 카드 등록 없이 바로 시작합니다. 연수 참여자 전원이 5분 안에 같은 출발선에 설 수 있다는 것은 연수 운영에서 결정적인 장점입니다.
2. **최신 Gemini 탑재.** 구글의 최신 AI 모델 Gemini가 게임을 생성합니다. 한국어 이해가 뛰어나 한국어 프롬프트만으로 충분하며, 생성 품질이 실습 만족도를 좌우합니다.
3. **배포까지 한 번에.** 만든 게임을 공유 링크로 즉시 배포하고, GitHub 양방향 동기화로 원본을 보관하며, Cloud Run으로 정식 서비스화할 수 있습니다. "만들고 끝"이 아니라 "학생에게 도달"까지 갑니다(5단원).
4. **Firebase 자동 연동.** 프롬프트만으로 Firestore 데이터베이스와 인증을 자동 구성합니다. 서버 지식 없이도 학급 전체의 점수가 모이는 실시간 리더보드를 만들 수 있습니다(6단원).

접속 주소는 aistudio.google.com입니다.[7] 3단원에서 실제 화면과 함께 만나게 되므로, 지금은 주소만 기억해 두면 됩니다.

2.8 AI Studio 기능 지도—앞으로 만날 여섯 무기

실습에서 사용할 여섯 기능의 지도입니다. 여섯 가지 모두 빌드(Build) 모드라는 하나의 화면에서 이루어집니다. 지금 외울 필요는 없습니다. 3단원부터 6단원까지 실제 화면 캡처와 함께 하나씩 익히게 되며, 여기서는 각 기능이 어느 단원에서 나오는지만 알아 두면 충분합니다.

- **프롬프트 앱 생성** —자연어 한 문단으로 게임의 뼈대가 완성됩니다. 실습의 시작점입니다(3단원).
- **갤러리 리믹스** —다른 사람이 만든 앱을 가져와 내 것으로 변형합니다. 처음부터 만들기 부담스러울 때 좋은 출발 자료가 됩니다.
- **주석 모드** —화면에서 고칠 곳을 직접 짚어 "여기 고쳐줘"라고 지시합니다. 말로 위치를 설명하기 어려울 때 쓰는 기능입니다.
- **Code 탭** —생성된 코드를 확인하고 수정합니다. 코드를 몰라도 실습에는 지장

[7] 무료 사용 조건, 배포 경로, Firebase 연동에 관한 서술의 근거는 다음과 같습니다. Google. (2026). *Build apps in Google AI Studio.* Gemini API 개발자 문서. https://ai.google.dev/gemini-api/docs/aistudio-build-mode; Google. (2026). *Deploying from Google AI Studio.* Gemini API 개발자 문서. https://ai.google.dev/gemini-api/docs/aistudio-deploying

이 없습니다.

- **GitHub 동기화** —내 저장소에 자동으로 업로드하고 양방향으로 맞춥니다. 출판의 통로입니다(5단원).
- **Cloud Run 배포** —클릭 한 번으로 실서비스 URL을 발급받습니다. 공유 링크의 상위 옵션에 해당합니다.

여섯 가지가 한 화면 안에 모여 있다는 점이 앞 절에서 말한 "일원화"의 실제 모습입니다.[8] 다른 도구를 쓴다면 생성은 이 서비스, 배포는 저 서비스, 데이터베이스는 또 다른 서비스로 화면을 옮겨 다녀야 합니다.

2.9 프롬프트 한 줄이 게임이 되는 순간

이론의 마무리는 눈으로 확인하는 것입니다. 연수라면 강사가 아래 프롬프트를 실제로 입력해 게임이 만들어지는 전 과정을 시연하고, 혼자 읽고 있다면 3단원에서 직접 입력해 보게 됩니다. 어느 쪽이든 중요한 것은 결과물이 아니라 그 사이에 벌어지는 일입니다.

```
초등 3학년이 분수 개념을 연습하는 낚시 게임을 만들어줘.
바다에 분수가 적힌 물고기들이 헤엄치고, 문제에 맞는 물고기를
클릭하면 낚아 올려. 정답이면 점수, 오답이면 물고기가 도망가.
밝고 귀여운 스타일로, 효과음도 넣어줘.
```

시연은 네 단계로 진행합니다

1. **프롬프트 입력.** 위 문장을 빌드 모드에 그대로 붙여넣습니다.
2. **생성 관찰.** 1–2분 동안 에이전트가 계획에서 코드, 실행으로 나아가는 과정을 지켜봅니다. 37쪽의 다섯 단계 루프가 실제로 도는 장면입니다.
3. **즉시 플레이.** 미리보기 화면에서 직접 낚시하며 결과를 확인합니다.

[8] 여섯 기능의 이름과 위치는 2026년 8월 화면 기준입니다. 실제 화면은 3–6단원의 캡처에서 확인할 수 있습니다. Google. (2026). *Build apps in Google AI Studio.* Gemini API 개발자 문서. https://ai.google.dev/gemini-api/docs/aistudio-build-mode

4. **수정 한 번.** "물고기를 더 천천히 움직여줘"라고 요청해 대화로 개선되는 과정을 봅니다.

함께 볼 세 가지 포인트

- **프롬프트 속 요소가 어떻게 구현되는가** —"분수가 적힌 물고기"라는 한 구절이 문제 출제 로직으로 번역되는 과정을 봅니다.
- **요청하지 않은 부분을 AI가 어떻게 채우는가** —배경, 점수판, 시작 화면처럼 말하지 않은 것들이 스스로 보완됩니다. 이 채워진 부분이 나중에 교사가 손볼 지점이 됩니다.
- **수정 요청 한 번에 무엇이 얼마나 바뀌는가** —대화형 개선(iteration)의 감각을 익히는 것이 이 시연의 진짜 목적입니다.

2.10 실습 로드맵—우리가 만들 것들

3단원부터는 매 단원 손에 잡히는 산출물이 하나씩 쌓입니다. 각 단원의 산출물이 다음 단원의 재료가 되는 누적 설계이므로, 건너뛰지 말고 순서대로 진행하기를 권합니다.

- **WORLD 3 · 나의 첫 퀴즈 게임** —AI Studio 화면을 익히고 프롬프트로 첫 게임을 생성합니다. 대화와 주석 모드로 수정하는 기본기를 다집니다.
- **WORLD 4 · 교과 맞춤 학습 게임** —내 과목과 단원, 학생 수준을 담은 본격 학습 게임을 만들고 점수·레벨·효과음 같은 게임 요소로 재미를 강화합니다.
- **WORLD 5 · 세상에 공개** —GitHub에 자동 업로드해 원본을 보관하고, 학생에게 URL과 QR코드로 배포합니다. 여러 출판 방법을 비교해 봅니다.
- **WORLD 6 · 실시간 리더보드** —Firebase를 연동해 학급 전체의 점수가 실시간으로 모이는 순위표를 붙입니다. 교실 전체가 함께 겨루는 게임의 완성입니다.

최종 결과물은 우리 반 교과 내용이 담긴, 링크 하나로 접속하는, 실시간 순위표가 도는 나만의 학습 게임입니다. 그리고 하나가 더 남습니다. 그것을 만들 줄 아는 교사 자신입니다.

2.11 시작 전 안전 수칙—네 개의 방패

AI로 만드는 교육 자료에는 네 가지 점검 지점이 있습니다. 실습 단원마다 해당 장면에서 다시 짚겠지만—배포에서는 개인정보를, 리더보드에서는 닉네임을—시작 전에 원칙으로 함께 확인해 둡니다.

- □ **저작권** —기존 게임의 캐릭터·이미지·음악을 그대로 가져다 쓰지 않습니다. AI가 생성한 이미지와 코드도 이용 조건을 확인하는 습관이 필요하며, 유명 IP를 닮은 생성물은 수업용이라도 피하는 것이 안전합니다.
- □ **개인정보** —게임 화면과 리더보드에 학생 실명이나 사진을 넣지 않습니다. 닉네임 사용을 원칙으로 하고, 수집하는 정보는 점수와 닉네임 정도로 최소화합니다. 6단원 리더보드 설계에서 필수 점검 사항입니다.[9]
- □ **학생 계정과 연령** —구글 계정 연령 정책과 소속 학교(교육청) 계정의 기능 제한을 사전에 확인합니다. 학생은 계정 없이 "만든 게임을 플레이"만 하도록 설계할 수 있으며, 이 책도 그 방식을 권장합니다.
- □ **AI 산출물 검토** —AI가 만든 문항의 사실 오류, 편향, 수준 적합성을 점검하는 것은 교사의 몫입니다. 교육 자료의 최종 책임자는 언제나 교사이며, 이 검토 안목이 에이전트 시대 교사 전문성의 핵심입니다.

> **교육청 지침이 우선입니다**
> 소속 시도교육청에 생성형 AI 활용 지침이 있다면 그것이 이 네 가지보다 우선합니다. 연수를 운영한다면 해당 지침을 함께 배부하기를 권합니다.

2.12 실습 준비 체크리스트

3단원 실습을 시작하기 전에 아래 다섯 항목을 점검합니다. 연수 운영자라면 이 목록을 사전 안내문으로 발송해 두는 것만으로 첫 실습 시간의 손실을 크게 줄일 수 있습니다. 특히 첫째 항목은 당일에 발견하면 되돌릴 방법이 마땅치 않습니다.

[9] 만 14세 미만 아동의 개인정보를 처리하려면 법정대리인의 동의가 필요합니다. 개인정보보호위원회. (2023). *개인정보 보호법 제22조의2(만 14세 미만 아동의 개인정보 보호).*; 교육부. (2024). *교육(지원)청·학교 개인정보 보호 가이드라인.*

- ☐ **구글 계정** —개인 계정을 권장합니다. 학교(워크스페이스) 계정은 관리자 설정에 따라 AI Studio 접속이 막혀 있을 수 있으므로, 사전에 확인하거나 개인 계정을 준비합니다.[10]
- ☐ **크롬 브라우저** —최신 버전으로 업데이트합니다. AI Studio는 크롬에서 가장 안정적으로 동작하며, 실습 중 브라우저 문제로 흐름이 끊기는 일을 예방합니다.
- ☐ **GitHub 계정 가입** —github.com에서 무료로 가입합니다. 5단원 온라인 출판에서 사용하며, 가입할 때 만든 사용자명이 게임 주소의 일부가 되므로 신중히 정합니다.
- ☐ **노트북과 안정적인 인터넷** —실습은 개인 노트북 기준으로 진행합니다. 태블릿은 일부 기능이 제한되며, 게임 생성 과정에서 안정적인 네트워크가 필요합니다.
- ☐ **접속 테스트** —aistudio.google.com에 로그인해 첫 화면이 열리는지만 미리 확인해 둡니다. 접속이 안 되는 경우 계정 유형의 연령·기관 제한을 점검합니다.

다섯 칸이 모두 채워졌다면 준비가 끝났습니다. 남은 것은 화면을 열고 첫 문장을 입력하는 일뿐입니다.

2.13 단원 마무리

이 단원에서 우리는 손을 움직이기 전에 알아 두어야 할 것들을 정리했습니다. 이제 3단원의 화면 앞에 앉을 준비가 되었습니다.

먼저 챗봇과 AI 에이전트의 차이를 작동 방식으로 갈랐습니다. 챗봇은 답을 주고 에이전트는 일을 끝냅니다. 목표 제시와 결과 검토가 사람의 몫으로 남는다는 것도 확인했습니다. 이어서 다섯 단계 루프로 에이전트가 스스로 반복하는 구조를 보았고, 교육 현장에서 이미 진행 중인 네 가지 활용 영역을 살펴보았습니다.

바이브 코딩에서는 마인드셋 하나를 챙겼습니다. 한국어로 설명하면 앱이 되고, 교사의 전문성인 교육 설계가 곧 개발력이 된다는 것입니다. 기술은 AI가 맡고 교육은 교사가 맡는 새로운 분업이 이 책의 전제입니다. 도구 세 가지를 비교한 뒤

[10] 계정 유형에 따른 이용 가능 여부와 연령 요건은 서비스 정책에 따라 달라집니다. Google. (2026). *Build apps in Google AI Studio.* Gemini API 개발자 문서. https://ai.google.dev/gemini-api/docs/aistudio-build-mode

Google AI Studio를 택한 이유도 정리했고, 빌드 모드·주석 모드·GitHub 동기화·Firebase 연동을 포함한 여섯 무기의 위치를 지도로 확인했습니다. 마지막으로 저작권·개인정보·학생 계정·AI 산출물 검토라는 네 방패를 세웠습니다. 최종 책임자는 언제나 교사입니다.

남은 숙제. 47쪽의 다섯 칸을 오늘 안에 모두 채워 두세요. 특히 aistudio.google.com 접속 테스트는 3단원을 펴기 전에 반드시 끝나 있어야 합니다. 그리고 내 교과에서 "게임으로 만들면 좋겠다" 싶은 단원을 하나만 골라 메모해 두세요. 4단원의 재료가 됩니다.

다음 단원 예고. WORLD 3 ·AI Studio 첫걸음—화면을 익히고 프롬프트 한 문단으로 첫 퀴즈 게임을 만듭니다. 대화와 주석 모드로 고치는 기본기까지 손에 붙입니다.

이론은 여기까지입니다. 다음 장부터는 만드는 시간입니다.

참고문헌

Karpathy, A. (2023). "The hottest new programming language is English" [게시물]. X.

Karpathy, A. (2025). "Vibe coding" [게시물]. X.

Russell, S., & Norvig, P. (2021). *Artificial intelligence: A modern approach* (4th ed.). Pearson.

Google. (2026). *Build apps in Google AI Studio.* Gemini API 개발자 문서. https://ai.google.dev/gemini-api/docs/aistudio-build-mode

Google. (2026). *Deploying from Google AI Studio.* Gemini API 개발자 문서. https://ai.google.dev/gemini-api/docs/aistudio-deploying

교육부. (2024). *교육(지원)청·학교 개인정보 보호 가이드라인.*

개인정보보호위원회. (2023). *개인정보 보호법 제22조의2(만 14세 미만 아동의 개인정보 보호).*

이 단원의 서비스 관련 서술(기능·배포 방식·요금·계정 조건)은 2026년 8월 기준입니다. 인공지능 서비스는 정책과 화면이 자주 바뀌므로 연수 직전에 각 서비스의 최신 안내를 확인하시기 바랍니다.

WORLD 3

AI Studio 첫걸음

접속부터 나의 첫 게임 완성까지

이 단원은 처음부터 끝까지 실습입니다. 브라우저 주소창에 주소를 입력하는 순간부터 로그인, 화면 구성 파악, 프롬프트 입력, 생성 관찰, 완성된 게임 플레이, 대화와 주석 모드로 고치기까지 모든 단계를 실제 화면과 함께 하나씩 따라갑니다.

오늘의 목표는 단 하나, “내 손으로 만든 첫 게임”입니다. 화면이 교재와 조금 달라도 흐름은 같습니다.

이 단원의 그림은 모두 실제 제작 과정을 그대로 촬영한 것입니다. 버튼의 위치나 이름은 서비스 업데이트로 달라질 수 있으니, 그림과 다르면 같은 뜻의 메뉴를 찾으시면 됩니다. 이름이 바뀌어도 역할은 같습니다.

이 단원의 도착점은 다음 한 줄입니다.

> **이 단원의 도착점**
> 「우리 몸과 건강 OX 퀴즈」—문제 10개, 정답마다 100점, 총점과 칭찬 메시지가 있는 게임 하나를 프롬프트 한 번으로 만들고, 대화로 고쳐 내 것으로 만듭니다.

3.1 오늘의 퀘스트와 실습 여정

3단원을 마치면 네 가지를 할 수 있어야 합니다. 이 단원의 퀘스트는 읽어서가 아니라 "직접 해보기"로만 달성됩니다. 그래서 설명을 줄이고 화면을 늘렸습니다. 지금 노트북을 열고, 그림을 옆에 두고 따라오시기 바랍니다.

첫째, AI Studio에 접속해 로그인하고 화면 구성을 설명할 수 있어야 합니다. 주소를 입력하고 로그인한 다음, 빌드 홈의 사이드바와 입력창과 갤러리가 각각 무엇을 하는 곳인지 파악하는 것이 여기에 해당합니다. 둘째, 프롬프트로 첫 게임을 생성할 수 있어야 합니다. 준비된 OX 퀴즈 프롬프트를 입력하고, 에이전트가 게임을 만들어 가는 과정을 눈으로 관찰합니다. 셋째, 대화와 주석 모드로 게임을 수정할 수 있어야 합니다. 문항 교체나 디자인 변경 같은 요청을 채팅으로, 그리고 화면을 직접 짚는 방식으로 전달합니다. 넷째, 내 교과 내용으로 OX 퀴즈를 변형할 수 있어야 합니다. 이것이 이 단원 마지막의 미션이며, 프롬프트 틀의 빈칸을 자기 과목과 단원으로 바꾸는 일입니다.

시작 전 점검

실습을 시작하기 전에 네 가지를 확인합니다. 2단원의 체크리스트를 이미 끝냈다면 그대로 통과됩니다.

- □ **크롬 브라우저 최신 버전** —자동 업데이트가 꺼져 있는 학교 노트북이 종종 있습니다.
- □ **구글 개인 계정 로그인 가능** —학교 계정이 아니라 개인 계정이어야 안전합니다.

- ☐ **노트북과 안정적인 인터넷** —생성 중에 연결이 끊기면 처음부터 다시 기다려야 합니다.
- ☐ **2단원 체크리스트 완료** —이 단원은 그 위에서 시작합니다.

여덟 개의 발판

오늘 밟을 순서는 여덟 단계입니다. 각 단계마다 실제 화면이 함께 제공되므로 자기 화면과 비교하며 따라오면 됩니다. 소요 시간은 여유 있게 잡아 총 60–80분입니다.

1. **접속.** aistudio.google.com을 주소창에 직접 입력합니다.[1]
2. **로그인.** 구글 계정으로 로그인하면 곧바로 빌드 홈에 도착합니다.
3. **화면 탐색.** 빌드 홈의 다섯 구역과 사이드바 메뉴를 파악합니다.
4. **입력창 확인.** 프롬프트 입력창과 게임을 시작하는 세 가지 방법을 봅니다.
5. **프롬프트 입력.** OX 퀴즈 프롬프트를 붙여넣고 Build를 실행합니다.
6. **생성 관찰.** 에이전트가 일하는 2–3분을 지켜봅니다.
7. **플레이.** 완성된 게임을 직접 해 보며 교사의 눈으로 점검합니다.
8. **수정.** 대화와 주석 모드로 고치고 다듬습니다.

중간에 길을 잃으면 72쪽의 트러블슈팅으로 가세요. 자주 막히는 여섯 지점의 해법이 정리되어 있습니다.

[1] Google. (2026). *Build apps in Google AI Studio.* Gemini API 개발자 문서. https://ai.google.dev/gemini-api/docs/aistudio-build-mode

3.2 접속과 로그인

크롬을 열고 주소창에 주소를 정확히 입력한 뒤 Enter를 누릅니다. 포털 검색으로 들어가면 광고나 이름이 비슷한 다른 사이트를 거칠 수 있으므로, 처음부터 주소를 직접 입력하기를 권합니다. 즐겨찾기에 바로 추가해 두면 다음 실습부터 편합니다.

그림 3-1. 크롬 주소창에 aistudio.google.com을 직접 입력한 모습(부분 캡처)

> **입력 팁**
>
> https://는 생략해도 됩니다. 자동으로 붙습니다.
>
> "ai studio"로 검색하지 말고 주소를 입력하세요. 이름이 비슷한 서비스가 여럿 있습니다.
>
> 주소창 오른쪽 별 모양 아이콘을 눌러 북마크에 추가해 둡니다.

주소를 입력하면 다음 화면이 나타납니다. 오른쪽 위의 Get started 버튼이 실습의 출발점입니다.

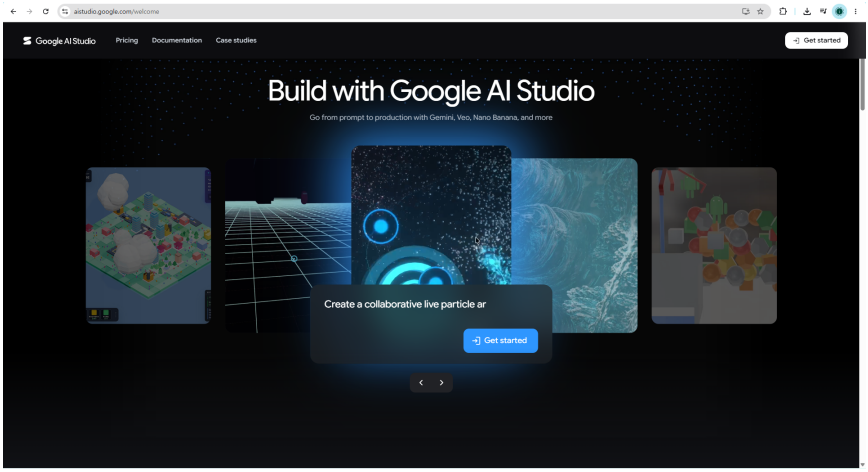


그림 3-2. 접속 직후 화면. 오른쪽 위의 Get started로 시작한다

> **왜 주소가 aistudio.google.com일까**
> AI Studio는 구글의 AI 개발 도구 사이트입니다. 구글 계정 하나로 로그인되며, 별도 설치 없이 브라우저에서 모든 실습이 이루어집니다. 접속 화면의 "Go from prompt to production with Gemini"라는 문구가 이 도구의 정체성을 요약합니다. 프롬프트에서 완성품까지라는 뜻입니다.

로그인—세 가지 경우의 수

접속 화면에서 오른쪽 위, 또는 가운데의 Get started 버튼을 누르면 로그인 절차가 시작됩니다. 자신의 상황이 아래 세 경우 중 어디에 해당하는지 먼저 확인하세요.

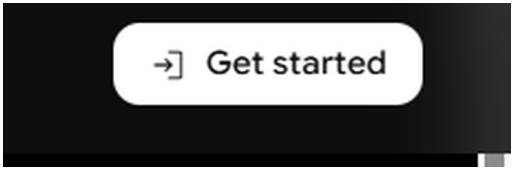


그림 3-3. 접속 화면 오른쪽 위의 Get started 버튼(부분 확대)

첫째, 이미 구글에 로그인되어 있는 경우입니다. 가장 흔합니다. Get started를 누르면 계정 확인 없이 곧바로 빌드 홈으로 들어갑니다. 크롬 오른쪽 위에 프로필 사진이 보인다면 이 경우입니다.

둘째, 로그인이 안 되어 있는 경우입니다. 구글 계정 선택 화면이 나타나므로 개인 계정을 선택하고 비밀번호를 입력하면 됩니다. 여러 계정이 있다면 실습용으로 정한 계정 하나만 계속 쓰는 편이 좋습니다. 계정이 섞이면 나중에 만든 게임을 찾기 어려워집니다.

셋째, 이 계정으로 AI Studio가 처음인 경우입니다. 이용 약관 동의 화면이 한 번 나타납니다. 내용을 확인하고 동의하면 이후에는 다시 나타나지 않습니다.

> **학교 계정이 안 될 때**
> 학교(워크스페이스) 계정은 관리자 설정에 따라 차단 안내가 뜰 수 있습니다. 개인 계정으로 다시 로그인하세요. 계정 전환은 화면 오른쪽 위 프로필 아이콘에서 합니다.

로그인에 성공하면 주소가 aistudio.google.com/apps로 바뀌며 다음 절의 화면

이 나타납니다.

3.3 빌드 홈 투어—다섯 구역만 알면 됩니다

로그인하면 별도의 이동 없이 곧바로 빌드 홈에 도착합니다. 가운데 “Build your ideas with Gemini”라는 문구 아래 프롬프트 입력창이 보이면 제대로 온 것입니다. 화면에 요소가 많아 보이지만 오늘 쓸 곳은 다섯 군데뿐입니다.

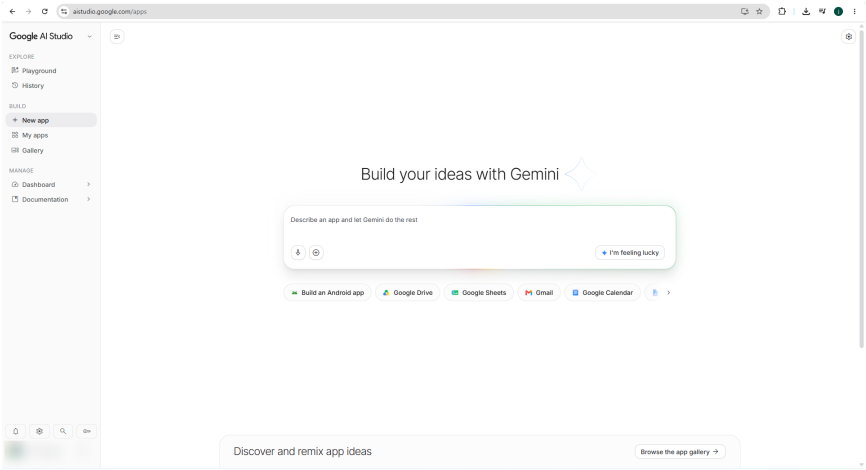


그림 3-4. 빌드 홈(aistudio.google.com/apps). 로그인 직후의 실제 화면

번호	구역	역할
1	사이드바	EXPLORE·BUILD·MANAGE 세 묶음의 기능 메뉴
2	History (기록)	모든 작업이 자동 저장. 실수로 닫아도 여기서 복구
3	프롬프트 입력창	게임 설명을 입력하는 시작점. 오늘의 주 무대
4	갤러리 입구	다른 사람들의 앱 구경. Browse the app gallery
5	계정·알림·설정	프로필 확인, 계정 전환, 환경 설정

메뉴 이름과 위치는 업데이트로 달라질 수 있습니다. 이름이 달라도 역할은 같으니, 표의 오른쪽 칸을 기준으로 자기 화면에서 다섯 구역을 직접 찾아보세요.

사이드바—메뉴 하나하나 뜯어보기

왼쪽 사이드바를 확대한 실제 화면입니다. 메뉴는 EXPLORE(탐색)·BUILD(만들기)·MANAGE(관리) 세 묶음으로 구성됩니다. 이 교재 실습의 주 무대는 BUILD 묶음이며, 그중에서도 New app이 새 게임을 만드는 시작 버튼입니다.

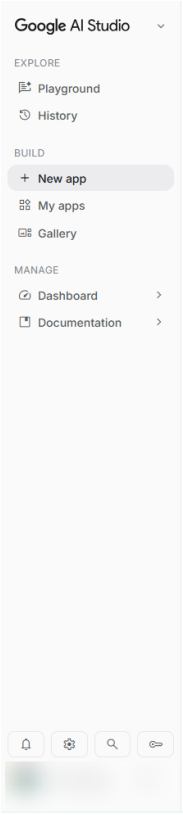


그림 3-5. 사이드바 부분 캡처. EXPLORE · BUILD · MANAGE 세 묶음

- **Playground**(EXPLORE) —Gemini 모델과 자유롭게 대화하고 실험하는 공간입니다. 게임에 넣을 문항 아이디어를 얻는 용도로 활용할 수 있습니다.
- **History**(EXPLORE) —지금까지의 작업이 자동 저장됩니다. 실수로 창을 닫아도 여기서 다시 열 수 있습니다.
- **New app**(BUILD) —새 앱, 곧 새 게임을 만드는 이번 실습의 시작점입니다. 빌드 홈의 입력창과 같은 기능입니다. 오늘의 주인공은 이 메뉴입니다.
- **My apps**(BUILD) —내가 만든 앱들의 목록입니다. 지난 실습에서 만든 게임을 다시 열 때 씁니다.
- **Gallery**(BUILD) —다른 사람들이 만든 앱을 구경하고 리믹스, 즉 복제해 수정할 수 있는 전시장입니다.
- **Dashboard · Documentation**(MANAGE) —API 키 등 개발자용 관리 메뉴

로, 이 교재 실습에서는 사용하지 않습니다.

사이드바가 아이콘만 보인다면 위쪽의 접기·펼치기 버튼으로 펼칠 수 있습니다.

3.4 입력창—게임을 시작하는 세 가지 방법

빌드 홈에서 게임 제작을 시작하는 방법은 세 가지입니다. 직접 프롬프트를 쓰는 것이 기본이고, 막막할 때는 나머지 두 방법이 좋은 출발점이 됩니다. 오늘은 첫째 방법, 직접 입력으로 첫 게임을 만듭니다.

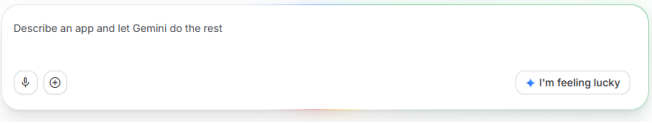


그림 3-6. 프롬프트 입력창(부분 확대). 오른쪽에 I'm feeling lucky 버튼이 보인다

첫째, 프롬프트 직접 입력

만들고 싶은 게임을 한국어로 설명하고 Build 버튼을 누릅니다. 오늘의 방법이며, 다음 절에 그대로 쓸 수 있는 프롬프트가 준비되어 있습니다. 다른 화면으로 갔다가 입력창으로 돌아오고 싶다면 사이드바의 New app을 누르면 됩니다.

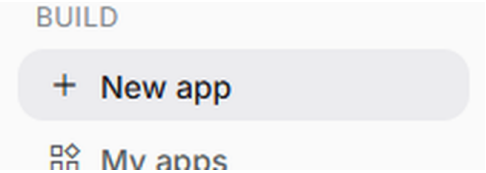


그림 3-7. 사이드바의 New app. 입력창으로 돌아오는 문

둘째, I'm feeling lucky

입력창 오른쪽의 버튼을 누르면 AI가 무작위 아이디어를 채워 줍니다. 무엇을 만들 수 있는지 감을 잡을 때 좋습니다. 채워진 문장을 그대로 실행해 보고, 마음에 드는 부분만 골라 자기 프롬프트에 옮겨도 됩니다.

셋째, 갤러리 리믹스

사이드바의 Gallery, 또는 빌드 홈 아래쪽의 Browse the app gallery에서 마음에 드는 앱을 골라 Remix하면 내 것으로 가져와 수정할 수 있습니다. 처음부터 만드는 것보다 빠르고, 잘 만든 게임의 구조를 뜯어보는 공부도 됩니다.

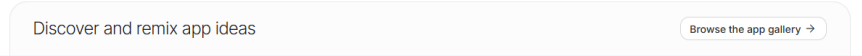


그림 3-8. 빌드 홈 아래쪽의 Discover and remix app ideas와 Browse the app gallery

Remix는 앱을 연 뒤에도 오른쪽 위에서 다시 만날 수 있습니다. 그 옆의 Share와 Publish는 만든 게임을 남에게 내보내는 버튼인데, 5단원에서 본격적으로 다룹니다. 지금은 이런 버튼이 있다는 것만 확인하고 넘어갑니다.

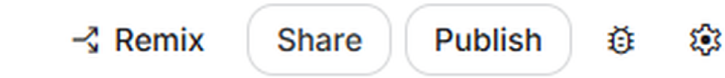


그림 3-9. 열린 앱 오른쪽 위의 Remix · Share · Publish

> **입력창 꿀팁—회색 글씨의 정체**
>
> 프롬프트를 쓰다 보면 뒤에 회색 글씨로 이어질 내용이 제안됩니다. Gemini의 자동 완성 제안으로, Tab 키를 누르면 채택되고 무시하면 반영되지 않습니다. 좋은 제안이면 받아들여 프롬프트를 풍성하게 만들 수 있습니다.

3.5 실습 프롬프트와 실행

아래 프롬프트를 그대로 입력하거나 배부된 파일에서 복사합니다. 그대로 써도 게임은 완성되지만, 이어지는 해설을 읽으면 "왜 이렇게 쓰는지"가 보입니다. 좋은 프롬프트는 대상·형식·규칙·스타일 네 요소를 담습니다.

```
초등학생용 OX 퀴즈 게임을 만들어줘.
주제는 우리 몸과 건강이야. 문제가 10개 나오고,
학생이 O 또는 X 버튼을 누르면 정답 여부를 알려줘.
정답이면 100점씩 올라가고, 다 풀면 총점과 함께
칭찬 메시지를 보여줘. 초등학생이 좋아할
밝고 알록달록한 디자인으로, 글자는 크게 해줘.
```

프롬프트의 네 요소

이 여섯 줄은 아무렇게나 늘어놓은 문장이 아닙니다. 1단원에서 배운 게임의 구성 요소가 그대로 들어가 있습니다.

요소	프롬프트의 표현	하는 일
대상	"초등학생용", "글자는 크게"	눈높이를 정합니다
형식	"OX 퀴즈", "문제 10개"	게임의 뼈대입니다
규칙	"정답이면 100점", "총점 표시"	1단원의 메커닉스에 해당합니다
스타일	"밝고 알록달록", "칭찬 메시지"	미학을 지정합니다

프롬프트에 정답을 다 적지 않아도 됩니다. 문항 자체는 AI가 만들어 줍니다. 대신 생성된 문항에 사실 오류가 없는지 확인하는 것은 교사의 몫입니다. 2단원의 안전 수칙 네 번째 항목이 여기에 그대로 적용되며, 65쪽에서 실제로 점검하게 됩니다.

입력하고 실행—Build 버튼 하나로

프롬프트를 입력창에 붙여넣은 실제 화면입니다. 캡처를 자세히 보면 프롬프트 끝에 회색 글씨의 자동 완성 제안("또한 캐릭터가 나와서 응원해 주는 애니메이션 효

과도 넣어주면 좋겠어.")이 붙어 있습니다. Tab을 누르면 이 제안까지 함께 채택됩니다.

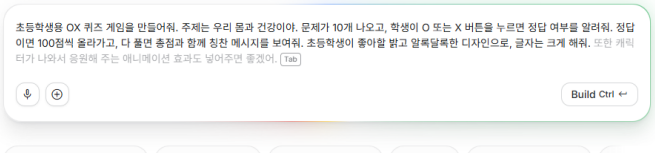


그림 3-10. 프롬프트가 입력된 입력창. 끝의 회색 글씨가 Gemini의 자동 제안

입력창 오른쪽 아래의 **Build** 버튼을 누르는 순간 에이전트가 일을 시작합니다. 단축키 Ctrl+Enter로도 실행되며, 입력창 안에서 줄바꿈은 Enter입니다.

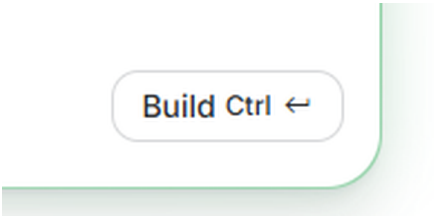


그림 3-11. Build 실행 버튼(부분 확대). 단축키 Ctrl+Enter가 함께 표시된다

실행 전 마지막 점검

프롬프트가 중간에 잘리지 않고 전부 들어갔는지 확인하세요. 복사와 붙여넣기 과정에서 마지막 문장이 빠지는 경우가 흔합니다. 일부만 입력되어도 게임은 만들어지지만, 빠진 요구사항(예를 들어 점수 규칙)은 반영되지 않습니다. 알림 권한 팝업이 뜨면 허용과 차단 어느 쪽을 눌러도 실습에는 지장이 없습니다.

3.6 생성 관찰—에이전트가 일하는 2 – 3분

실행하면 화면이 작업 모드로 바뀌고 에이전트의 진행 상황이 실시간으로 표시됩니다. 이 2 – 3분이 2단원에서 배운 "에이전트 루프"가 눈앞에서 돌아가는 순간입니다. 기다리는 동안 자리를 뜨지 말고 왼쪽 목록이 어떻게 늘어나는지 지켜보세요.

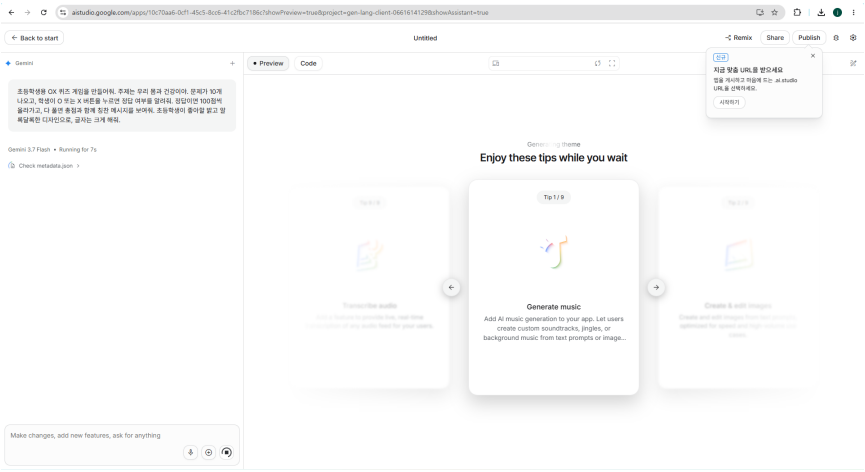


그림 3-12. 생성 중 화면. 왼쪽에 사용 모델과 경과 시간(Running for …s)이 표시된다

화면은 크게 두 쪽으로 나뉩니다. 왼쪽에는 진행 상황이 나옵니다. 사용 중인 모델과 경과 시간이 표시되고, 파일이 하나씩 만들어질 때마다 목록이 늘어납니다. 2단원에서 나눈 에이전트 루프의 계획 단계와 도구 사용 단계가 이 목록으로 드러나는 셈입니다. 오른쪽에는 대기 중에 활용 팁 카드(“Enjoy these tips while you wait”)가 넘어갑니다. 가볍게 읽어 두면 4단원부터 6단원까지의 실습에서 도움이 됩니다.

생성이 끝나면 오른쪽에 게임 미리보기가 자동으로 열립니다. 에이전트 루프의 실행과 생성 단계가 마무리된 것이고, 보통 2–3분 안에 완료됩니다.

> 생성이 5분 이상 멈춰 있으면 72쪽 트러블슈팅의 세 번째 항목을 보세요. 대부분 새로고침 없이 해결됩니다.

3.7 완성—첫 게임을 플레이하고 점검하기

첫 게임이 완성되었습니다. 실제 생성 결과를 보면 요청한 10문제와 100점 규칙은 물론, 요청하지 않은 소리로 듣기(TTS) 기능과 진행률 표시까지 에이전트가 스스로 더해 놓았습니다. 미리보기에서 직접 플레이하며 교사의 눈으로 점검하세요.

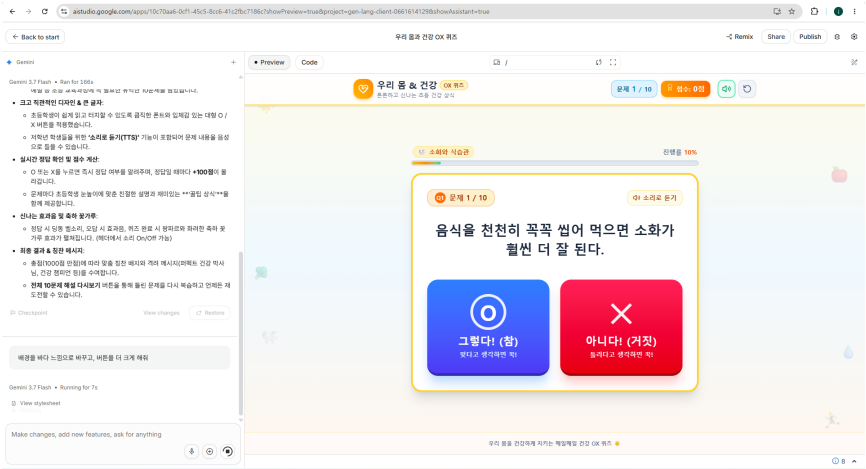


그림 3-13. 완성된 게임 플레이 화면. 문제 1/10, O와 X 버튼, 점수와 진행률 표시

플레이하며 점검할 다섯 가지

지금 하는 점검은 감상이 아니라 다음 단계의 재료입니다. 마음에 들지 않는 점을 두세 개 메모해 두면, 바로 다음 절에서 그것을 고치게 됩니다.

- ☐ **요구가 다 반영되었는가** —10문제, 100점, 총점, 칭찬 메시지가 모두 들어 있는지 확인합니다.
- ☐ **문항 내용이 정확한가** —사실 오류나 어색한 표현이 없는지 봅니다. 이 확인은 교사만 할 수 있습니다.
- ☐ **난이도가 맞는가** —우리 반 학생 수준에 맞는지 판단합니다.
- ☐ **버튼이 제대로 눌리는가** —O와 X 버튼의 반응이 명확한지 직접 눌러 봅니다.
- ☐ **교실 화면에서 보이는가** —글자 크기와 색이 TV나 빔프로젝터에서도 잘 보일

지 가능합니다.

3.8 대화로 수정하기—개발자와 회의하듯

수정은 새 프롬프트가 아니라 이어지는 대화로 합니다. 왼쪽 채팅창에 원하는 변경을 말하면 에이전트가 기존 게임을 고쳐 줍니다. 처음부터 다시 만드는 것이 아니라, 지금 있는 게임을 손보는 것입니다.

그림 3-14. 수정 요청을 입력한 채팅창(부분)

무엇을 어떻게 요청할지 막막하다면 아래 여섯 가지 유형을 참고하세요. 실제 연수에서 가장 자주 쓰인 요청들입니다.

유형	요청 문장의 예
문항 교체	“3번 문제를 뼈와 근육에 관한 문제로 바꿔줘”
난이도 조절	“문제를 조금 더 쉽게, 5학년 수준으로 바꿔줘”
디자인 변경	“배경을 바다 느낌으로, 버튼을 더 크게 해줘”
효과 추가	“정답일 때 박수 소리와 폭죽 효과를 넣어줘”
규칙 변경	“제한 시간 10초 타이머를 문제마다 추가해줘”
한국어 다듬기	“문장을 초등학생에게 더 친근한 말투로 바꿔줘”

요청 하나가 화면 전체를 바꿉니다

“배경을 바다 느낌으로 바꾸고, 버튼을 더 크게 해줘”라는 한 문장의 결과입니다. 배경만 바뀐 것이 아니라 제목과 캐릭터, 아이콘까지 바다 테마로 일관되게 다시 디자인된 점에 주목하세요. 게임 이름도 “바다 속 우리 몸과 건강 OX 퀴즈 대탐험”으로 바뀌었습니다.

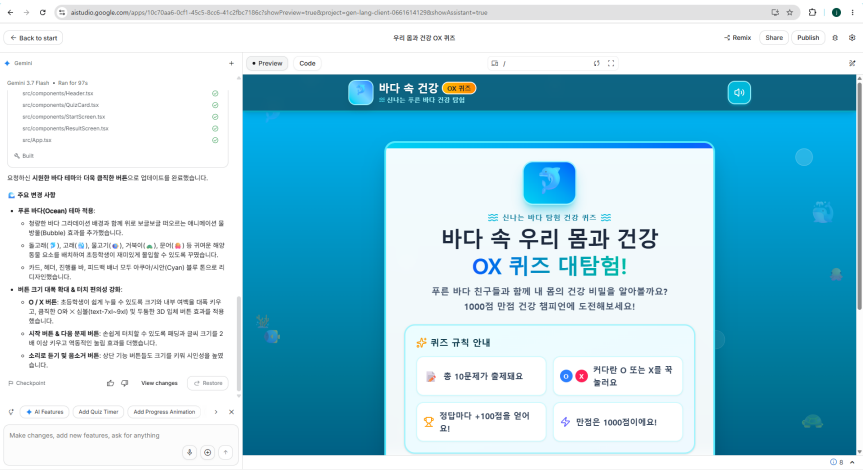


그림 3-15. 수정 결과. 요청 한 줄로 제목·캐릭터·색까지 바다 테마로 다시 디자인되었다

요령

한 번에 하나씩, 구체적으로 요청합니다. 세 가지를 한꺼번에 말하면 어느 것이 반영되고 어느 것이 빠졌는지 확인하기 어렵습니다. 되돌리고 싶다면 “방금 수정 취소하고 원래대로”라고 말하면 됩니다.

3.9 주석 모드—말 대신 화면을 짚어서

"오른쪽 위의 그거"를 말로 설명하기 어려울 때가 있습니다. 미리보기 오른쪽 위의 연필 모양 아이콘을 누르면 주석 모드가 켜집니다. 고칠 곳을 직접 클릭해 지목하는 방식이라, 디자인 세부 수정에서 특히 강력합니다.

그림 3-16. 미리보기 오른쪽 위의 연필 아이콘(부분)

1. 연필 아이콘을 눌러 주석 모드를 켭니다.
2. 고칠 요소를 클릭합니다. 파란 테두리로 선택 표시가 됩니다.
3. 요청을 입력합니다. 선택한 요소의 이름표가 채팅창에 함께 붙어 전달됩니다.
4. Submit을 누르면 해당 부분만 정확히 수정됩니다.

아래는 제목 요소를 선택한 모습입니다. 선택된 영역에 파란 테두리가 생기고, 요소의 이름표(h1 Component)가 채팅창 아래에 붙었으며, 화면 하단에는 선택과 형광펜, 색상 등을 다루는 주석 도구 막대가 열렸습니다.

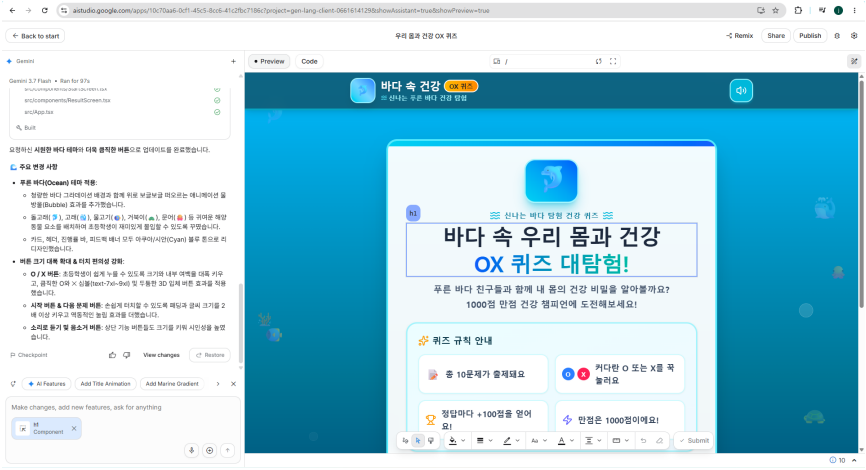


그림 3-17. 제목(h1) 요소를 선택한 모습. 이름표가 채팅창에 붙고 하단에 주석 도구 막대가 열린다

활용 예를 들면 “이 버튼만 노란색으로”, “이 글자를 두 줄로 나눠줘” 같은 요청입니다. 말로 설명하면 길고 애매한 것들이 클릭 한 번으로 명확해집니다.

3.10 Code 탭 구경과 이름 붙이기

미리보기 옆의 Code 탭을 누르면 에이전트가 작성한 실제 코드가 보입니다. 읽지 못해도 전혀 상관없습니다. “내 한국어 문장이 이만큼의 코드로 번역되었구나”를 확인하는 것으로 충분합니다.

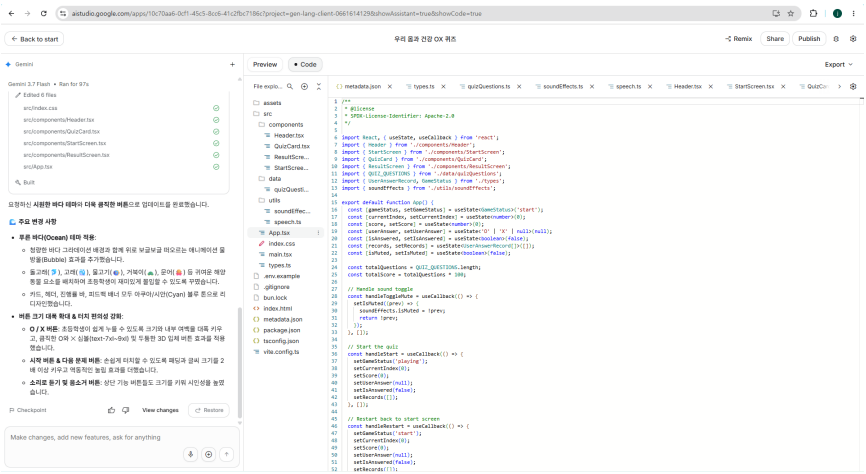


그림 3-18. Code 탭. 파일 목록과 코드. 우리가 쓴 프롬프트의 번역본이다

파일이 역할별로 정리되어 있다는 점만 눈여겨보세요. 화면을 구성하는 조각들은 components 폴더에, 문항 데이터는 quizQuestions에, 효과음은 soundEffects에 들어 있습니다. 나중에 게임에 문제가 생겼을 때 “문항이 이상하다”보다 “문항 데이터를 고쳐 달라”고 말할 수 있게 되는 것, 그것이 이 구경의 목적입니다.

이름 확인과 변경

AI가 알아서 "우리 몸과 건강 OX 퀴즈"라는 이름을 붙여 주었습니다. 화면 위 가운데의 제목을 클릭하면 직접 바꿀 수 있습니다.

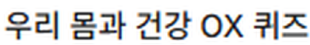


그림 3-19. 화면 위 가운데에 표시된 프로젝트 이름

이름 규칙을 하나 제안합니다. 「과목_주제_게임형식」 형태로, 예를 들면 과학_우리몸_OX퀴즈입니다. 작업물이 한두 개일 때는 규칙의 필요를 못 느끼지만, 한 학기가 지나 열 개가 넘어가면 규칙의 힘이 커집니다.

작업은 자동 저장되며, 사이드바의 My apps와 History에서 언제든 다시 열 수 있습니다. 창을 잘못 닫아도 잃어버리지 않으니 안심하고 이것저것 시도해 보세요.

3.11 20분 미션—내 교과의 OX 퀴즈로

이제 배운 것을 내 것으로 만들 시간입니다. 아래 프롬프트 틀의 빈칸을 자신의 학년과 과목, 단원으로 채워 새 게임을 만들어 보세요. 완성하면 옆 사람과 서로의 게임을 플레이하고 한 가지씩 칭찬과 제안을 나눕니다. 20분을 잡습니다.

```
[ ]학년용 OX 퀴즈 게임을 만들어줘.
주제는 [ 과목의 단원]이야.
문제가 [ ]개 나오고, 정답이면 [ ]점씩 올라가.
[우리 반 아이들이 좋아할 스타일 설명]으로 꾸며줘.
글자는 교실 TV에서도 잘 보이게 크게 해줘.
```

1. **빈칸을 채워 새 게임을 생성합니다.** 사이드바의 New app으로 새로 시작하세요. 기존 게임의 대화에 이어 쓰면 앞의 게임이 덮어써집니다.
2. **생성된 문항을 검토하고 수정합니다.** 사실 오류 점검은 교사의 몫입니다. 틀린 문항은 대화로 다듬습니다(66쪽).

3. **짝과 교차 플레이를 합니다.** 서로의 게임을 해 보고 칭찬 하나와 제안 하나를 나눕니다.
4. **이름을 붙여 저장을 확인합니다.** 「과목_주제_OX퀴즈」 규칙으로 이름을 바꾸고, My apps에서 보이는지 확인합니다.

> **시간이 남는다면**
> "문제를 쉬움 5개와 어려움 5개로 나눠줘", "마지막에 틀린 문제만 다시 풀게 해줘" 같은 요청으로 게임을 발전시켜 보세요. 4단원에서 이 게임을 더 크게 키웁니다.

3.12 트러블슈팅—막히는 여섯 지점

실습 중 자주 발생하는 상황과 해결 순서입니다. 대부분 1분 안에 해결됩니다. 연수 운영자는 이 절을 펼쳐 두고 진행하면 좋습니다.

Q. 접속했는데 로그인 화면이 안 보여요

A. 이미 다른 구글 계정으로 로그인된 상태일 수 있습니다. 오른쪽 위 프로필 아이콘을 누르고 계정 전환에서 원하는 계정을 선택하세요(55쪽).

Q. 학교 계정이라 차단된대요

A. 개인 계정으로 전환하세요. 2단원 체크리스트에서 다룬 내용입니다. 크롬 프로필을 개인용으로 새로 만들면 계정이 섞이지 않습니다.

Q. 생성이 5분 넘게 멈춰 있어요

A. 채팅창에 "계속해줘"라고 입력해 보세요. 그래도 멈춰 있으면 새로고침(F5)합니다. 작업은 자동 저장되어 사라지지 않습니다.

Q. 게임에 오류가 나요 / 버튼이 안 눌려요

A. 오류 화면을 그대로 두고 채팅에 "오류가 났어, 고쳐줘"라고 입력하세요. 에이전트가 스스로 원인을 찾아 수정합니다.

Q. 만들던 게임이 사라졌어요

A. 사이드바의 My apps 또는 History를 여세요. 모든 작업이 자동 저장되어 있습니다. 프로젝트 이름을 붙여 두면 찾기 쉽습니다(70쪽).

Q. 문항 내용이 이상해요

A. 정상입니다. AI는 사실 오류를 낼 수 있습니다. "3번 문제가 틀렸어. ○○로 바꿔줘"라고 구체적으로 수정을 지시하세요.

3.13 단원 마무리

첫 게임이 완성되었습니다. 오늘 한 일은 프로그래밍이 아니라 "요청하기"였습니다. 그 요청이 어떤 형태여야 원하는 결과가 나오는지, 이제 손으로 알게 되었습니다.

이 단원에서 우리는 주소를 직접 입력해 AI Studio에 접속했고, 로그인 세 경우를 구분해 각자의 상황에 맞게 들어갔습니다. 빌드 홈의 다섯 구역과 사이드바의 세 묶음을 익혀 화면에서 길을 잃지 않게 되었고, 대상과 형식, 규칙, 스타일 네 요소를 담은 프롬프트 하나로 OX 퀴즈 게임을 만들어 2–3분의 생성 과정을 지켜보았습니다. 완성된 게임을 교사의 눈으로 다섯 가지 항목에 따라 점검했고, 대화로 고치기와 주석 모드로 화면 짚기라는 두 가지 수정 기술을 손에 익혔습니다. Code 탭에서 내 한국어 문장의 번역본을 확인했고, 마지막으로 내 과목과 단원의 OX 퀴즈를 만들어 이름을 붙여 저장했습니다.

남은 숙제. 미션에서 만든 게임을 다음 수업에서 5분만 써 보고, 학생 반응을 한 문장으로 기록해 두세요. 어느 문항에서 웃었고 어디에서 막혔는지가 4단원 설계의 재료가 됩니다.

다음 단원 예고. WORLD 4 · 교과 맞춤 학습 게임 만들기—OX 퀴즈보다 한 단계 큰 게임으로 넘어갑니다. 단원 목표에서 출발해 게임 형식을 고르고, 여러 차례 대화로 다듬어 실제 수업에 쓸 수 있는 자료를 완성합니다.

오늘 만든 게임은 아직 내 화면 안에만 있습니다. 밖으로 내보내는 방법은 5단원에서 다룹니다.

참고문헌

Google. (2026). *Build apps in Google AI Studio.* Gemini API 개발자 문서. https://ai.google.dev/gemini-api/docs/aistudio-build-mode

Google. (2026). *Google AI Studio.* https://aistudio.google.com

화면 캡처: Google AI Studio(aistudio.google.com), 2026년 8월. 이 단원의 캡처는 실제 제작 과정을 그대로 촬영한 것이며, 서비스 정책에 따라 화면 구성은 변경될 수 있습니다.

WORLD 4

교과 맞춤 학습 게임

퀴즈를 넘어 내 수업을 담은 게임으로

3단원에서 우리는 “게임을 만들 수 있다”는 것을 확인했습니다. 이번 단원의 질문은 한 단계 위입니다—“좋은 학습 게임을 만들 수 있는가.”

1단원의 MDA를 기획 언어로 꺼내어 학습 목표를 게임 규칙으로 번역하고, 내 교과에 맞는 장르를 골라 본격 게임을 만듭니다. 단원의 끝에서 선생님의 게임은 “퀴즈”에서 “우리 반 수업 도구”가 됩니다.

이 단원은 앞 단원들과 성격이 다릅니다. 3단원이 "버튼을 어디에서 누르는가"를 다뤘다면, 이 단원은 "무엇을 만들 것인가"를 다룹니다. 화면 조작은 이미 배운 것에서 크게 늘어나지 않는 대신, 판단해야 할 것이 훨씬 많아집니다. 같은 도구로도 기획이 다르면 전혀 다른 수업 도구가 나오기 때문입니다.

그래서 이 단원은 그림이 적습니다. 대신 종이에 쓰고, 고르고, 대조하는 시간이 깁니다. 본문의 그림은 2026년 8월 Google AI Studio에서 촬영한 것이며, 서비스 정책에 따라 버튼의 위치나 이름이 바뀔 수 있으니 그림과 다르면 같은 뜻의 메뉴를 찾으시면 됩니다.

> **이 단원의 도착점**
>
> 「과목_주제_장르」로 이름 붙인 나의 대표 게임 한 편.
>
> 내 교과 내용이 담겨 있고, 게임 요소가 두 개 이상 작동하고, 배려 장치가 하나 이상 들어 있고, 전 문항을 교사가 검수한 게임입니다. 이 게임은 5단원에서 학생에게 배포되고, 6단원에서 실시간 리더보드를 달게 됩니다.

4.1 이 단원의 퀘스트와 여정

이번 단원의 핵심은 기술이 아니라 설계입니다. 프롬프트를 잘 치는 능력이 결과를 가르는 것이 아니라, 무엇을 만들지 정하는 능력이 결과를 가릅니다. 먼저 도착점 네 가지와 그곳까지 가는 여섯 단계를 펼쳐 놓고 시작하겠습니다.

네 가지 퀘스트

이 단원을 마치면 다음 네 가지를 할 수 있어야 합니다. 순서대로 쌓이는 구조라서, 앞의 것을 건너뛰면 뒤의 것이 헐거워집니다.

- **학습 목표를 게임 규칙으로 번역하는 미니 기획서를 쓸 수 있다.** 성취기준에서 출발해 핵심 행동, 게임 메커닉(M), 노리는 재미(A)까지의 연결을 한 장으로 정리합니다.
- **내 교과에 맞는 장르를 고르고 본격 게임을 생성할 수 있다.** 장르 카탈로그 여섯 가지 중에서 교과 특성에 맞는 형식을 골라 프롬프트로 구현합니다.
- **게임 요소로 재미를 의도적으로 강화할 수 있다.** 요소 메뉴판에서 두셋을 골라 추가하고, 각 요소가 어떤 재미를 만드는지 설명할 수 있습니다.

- **난이도 곡선과 배려 설계, AI 실시간 기능으로 게임을 완성한다.** 쉬움에서 어려움으로 가는 배치, 다시 풀기, 저학년 배려를 더하고 Gemini 실시간 기능까지 실험합니다.

여섯 단계 여정

이 단원은 설계에서 제작으로, 제작에서 강화와 완성으로 흐릅니다. 소요 시간은 여유 있게 90–120분을 잡으시면 됩니다. 각 단계에서 내리는 교육적 결정이 게임의 품질을 좌우하므로, 서두르지 않는 편이 결국 빠릅니다.

단계	하는 일	무대
1 미니 기획서	학습 목표를 게임 규칙으로 번역합니다(MDA 실전)	종이
2 장르 선택	교과 특성에 맞는 게임 형식을 고릅니다	종이
3 본격 게임 생성	기획서를 프롬프트로 바꿔 새 게임을 만듭니다	AI Studio
4 게임 요소 강화	타이머·콤보·레벨 등 재미 장치를 두 셋 더합니다	AI Studio
5 난이도·배려 설계	쉬움에서 어려움으로, 다시 풀기, 저학년 배려	AI Studio
6 플레이테스트	교사·학생 두 시점 점검과 동료 피드백 루틴	교실

> **진행 팁**
> 1–2단계는 종이에서, 3–6단계는 AI Studio에서 이루어집니다. 전체 시간의 3분의 1을 설계에 쓰는 것이 결과적으로 가장 빠른 길입니다. 기획서 없이 프롬프트부터 치면 대화가 길어지고, 길어진 대화는 대체로 원래 만들려던 것에서 멀어집니다.

이 단원의 재료

3단원에서 만든 나의 OX 퀴즈를 업그레이드 재료로 쓸 수도 있고, 완전히 새 장르로 갈아탈 수도 있습니다. 어느 쪽이든 순서는 같습니다. 기획서가 먼저입니다. 이미 만든 퀴즈가 있다면 그것을 기획서 칸에 거꾸로 채워 보는 것부터 해 보셔도

좋습니다. 대개 ②번 칸에서 막히는데, 그 막힘이 바로 이 단원이 다루려는 지점입니다.

4.2 미니 기획서—학습 목표를 게임 규칙으로 번역하기

프롬프트를 치기 전에 종이 한 장을 먼저 씁니다. 1단원에서 배운 MDA를 실전에 투입하는 순간입니다. 네 칸을 채우면 그것이 곧 프롬프트의 뼈대가 되고, 나중에 만들어진 게임을 판정하는 기준표가 됩니다. 칸은 넷뿐이고, 각 칸은 한두 줄이면 충분합니다.

① 학습 목표·성취기준—어디에 도달시킬 것인가

이 게임으로 무엇을 배우게 할 것인지를 교육과정의 성취기준 한 줄로 시작합니다. 여기서 자주 나오는 실수는 범위를 너무 넓게 잡는 것입니다. "분수를 이해한다"는 게임 한 편이 감당할 수 없습니다. "단위분수와 진분수의 크기를 비교할 수 있다" 정도로 좁혀야 다음 칸이 써집니다. 한 게임에 성취기준 하나, 이것이 원칙입니다.

② 핵심 행동—학생이 게임 속에서 반복할 동작

학생이 이 게임에서 수십 번 반복하게 될 행동은 무엇인가. 고르기, 맞히기, 모으기, 막기, 배치하기, 잇기 가운데 하나로 적어 봅니다. 이 칸이 미니 기획서의 심장입니다. 핵심 행동이 학습 행동과 겹칠수록 좋은 학습 게임이 되고, 어긋날수록 "문제 사이에 오락이 끼어든 것"이 됩니다.

③ 게임 메커닉(M) —점수와 실패의 규칙

핵심 행동을 담을 규칙을 씁니다. 무엇을 하면 점수가 오르고, 무엇을 하면 실패하며, 어디에 도달하면 한 판이 끝나는가. 이 세 가지가 정해지면 게임은 이미 절반 만들어진 셈입니다. 점수 규칙은 학생이 무엇을 열심히 하게 될지를 결정하므로, 학습적으로 중요한 행동에 점수가 붙어 있는지 확인해야 합니다.

④ 노리는 재미(A) —여덟 중에서 두셋

1단원의 미학 8분류 가운데 무엇을 노릴지 고릅니다. 도전을 축으로 두셋을 고르는 것이 요령입니다. 하나뿐이면 단조롭고, 넷을 넘으면 초점이 흐려집니다. 이 칸은 나중에 게임 요소를 고를 때 그대로 판단 기준이 됩니다. 수집의 재미를 노렸다면 어항이나 도감 같은 장치가 들어가야 하고, 감각의 재미를 노렸다면 효과음과 화면 연출에 요청을 써야 합니다.

작성 예시—수학 3–4학년 「분수」

네 칸을 실제로 채우면 다음과 같습니다. 이 예시는 이어지는 절에서 프롬프트로, 그다음 절에서 점검표로 계속 쓰이므로 눈에 익혀 두시기 바랍니다.

칸	내용
① 목표	단위분수·진분수의 크기를 비교할 수 있다
② 핵심 행동	두 분수 중 큰 쪽을 빠르게 고르기
③ 메커닉	바다에 분수가 적힌 물고기 두 마리가 헤엄침. 큰 분수를 낚으면 +100점, 작은 쪽을 낚으면 물고기가 도망감. 20마리 중 15마리 이상 잡으면 클리어
④ 재미	도전(비교 판단) + 수집(어항이 채워짐) + 감각(낚는 손맛)

이 네 줄이 그대로 다음 절의 프롬프트가 됩니다. 기획서를 쓴다는 것은 결국 프롬프트를 미리 써 두는 일입니다.

> **기획서를 판정하는 한 가지 질문**
>
> ②번 칸과 학습 행동이 겹치는가. 이것이 "분수를 비교하는 게임"인지 "분수 문제 사이에 끼어든 오락"인지를 가릅니다. 문제를 맞히면 캐릭터가 점프하는 게임에서 학생이 실제로 반복하는 것은 점프 타이밍 맞추기일 수 있습니다. 그것이 나쁘다는 뜻은 아니지만, 그 게임을 20분 해도 분수 실력은 늘지 않는다는 사실은 알고 써야 합니다.

연수용 양식

연수나 동학년 협의회에서는 다음 양식을 한 장씩 나눠 쓰면 10분이면 채워집니다. 칸을 다 채우지 못해도 괜찮습니다. 비어 있는 칸이 곧 아직 정하지 못한 결정이고, 그 결정을 AI가 대신 내리게 되는 지점입니다.

미니 기획서 양식

① 학습 목표·성취기준

_____

② 핵심 행동 (학생이 반복할 동작 하나)

_____

③ 게임 메커닉

점수 _____ 실패 _____

클리어 조건 _____

④ 노리는 재미 (도전을 축으로 2–3개)

☐ 도전 ☐ 수집 ☐ 감각 ☐ 발견

☐ 친교 ☐ 서사 ☐ 표현 ☐ 환상

한 문장 정리—"이 게임은 ____을(를) 반복시켜 ____에 도달한다"

_____

4.3 장르 카탈로그—교과가 형식을 고른다

AI 에이전트는 어떤 장르든 만들 수 있습니다. 그러므로 선택 기준은 기술이 아니라 교과 특성입니다. 반복 연습이 필요한 내용인가, 개념 사이의 관계를 다루는가, 탐구 과정 자체가 목표인가에 따라 어울리는 형식이 다릅니다. 아래 여섯 가지를 훑고 하나를 고르시면 됩니다.

반복 연습이 필요한 내용이라면

연산, 어휘, 구구단, 낱말처럼 정확도와 속도를 함께 길러야 하는 내용에는 수집형과 타워 디펜스가 어울립니다. 수집형은 반복 자체를 모으는 재미로 바꿔 줍니다. 정답을 맞힐 때마다 물고기가 어항에, 작물이 창고에 쌓이면 학생은 같은 유형의 문제를 스무 번 푸는 일을 지루해하지 않습니다. 타워 디펜스는 반복을 자동화 훈련으로 바꿉니다. 문제를 맞혀야 포탑이 발사되므로, 정확하게 그리고 빠르게 답해야 한다는 압력이 자연스럽게 생깁니다.

개념 사이의 관계를 다룬다면

사회·역사·과학처럼 개념끼리의 순서, 인과, 분류가 핵심인 내용에는 카드 배틀이 어울립니다. 카드에 인물이나 사건, 물질을 얹고 시대 순서나 강약 관계로 대결하게 하면, 학생은 개별 사실을 외우는 대신 사실들 사이의 관계를 다루게 됩니다. 단순 암기와 관계 이해를 가르는 지점이 여기입니다.

탐구 과정 자체가 목표라면

통합 단원이나 프로젝트 학습, 추론이 중심인 수업에는 방탈출이 어울립니다. 단서 세 개를 순서대로 풀어야 문이 열리는 구조는 학생에게 "무엇을 먼저 확인해야 하는가"를 생각하게 만듭니다. 다만 제작과 검수에 시간이 가장 많이 드는 형식이므로, 첫 게임으로는 권하지 않습니다.

저학년이거나 처음이라면

저학년에게는 러너·점프가 잘 맞습니다. 신체 리듬에 간단한 문제를 얹는 형식이라 규칙 설명이 짧고, 읽기 부담이 적습니다. 그리고 어느 학년이든 처음 만드는 게임이라면 퀴즈 대전이 가장 안전한 출발입니다. 구조가 단순해 AI가 실패할 확률이 낮고, 실패해도 어디가 잘못됐는지 금방 보입니다.

여섯 장르와 프롬프트 씨앗

각 장르의 프롬프트 씨앗은 기획서 내용과 합치면 그대로 실습 프롬프트가 됩니다.

장르	어울리는 교과	프롬프트 씨앗
퀴즈 대전	전 교과 복습. 가장 안전한 출발	"○○ 단원 4지선다 퀴즈 대전, 연속 정답 콤보"
수집형	수학 연산·어휘. 반복을 수집의 재미로	"정답을 맞히면 물고기를 낚아 어항에 모으는 게임"
타워 디펜스	구구단·낱말 등 자동화 훈련	"문제를 맞혀야 포탑이 발사되는 낱말 방어 게임"
카드 배틀	사회·역사·과학 개념 관계	"역사 인물 카드로 시대 순서 대결하는 게임"
방탈출	통합·프로젝트. 추론 과정	"단서 3개를 풀어야 문이 열리는 과학 방탈출"
러너·점프	저학년. 신체 리듬 + 간단 문제	"장애물 앞 문제를 맞혀야 점프하는 달리기 게임"

> **선택 기준 한 줄**
> 반복 연습이면 수집형·디펜스, 개념 관계면 카드 배틀, 탐구 과정이면 방탈출, 처음이면 퀴즈 대전.

4.4 본격 게임 생성—기획서를 프롬프트로

기획서의 네 칸을 문장으로 잇기만 하면 프롬프트가 됩니다. 새로 쓰는 것이 아니라 옮겨 적는 것입니다. 시작하기 전에 사이드바의 New app으로 새 프로젝트를 여세요. 3단원 게임의 대화에 이어 쓰면 두 게임이 섞입니다.

실습 프롬프트—기획서를 문장으로

앞 절의 분수 기획서를 그대로 옮기면 다음과 같습니다.

```
초등 3--4학년 분수 비교 낚시 게임을 만들어줘.
바다에 분수가 적힌 물고기 두 마리가 헤엄치고,
더 큰 분수 물고기를 클릭하면 낚아서 +100점,
작은 쪽을 클릭하면 물고기가 도망가.
낚은 물고기는 화면 아래 어항에 모이고,
20마리 중 15마리를 잡으면 스테이지 클리어야.
바다 느낌의 시원한 디자인, 글자는 크게 해줘.
```

문장 어디가 기획서의 어느 칸인지 짚어 보면 구조가 보입니다.

- **첫 문장—기획서 ①·②** 대상 학년과 과목, 그리고 핵심 행동(비교해 고르기)을 한 줄에 담습니다.
- **가운데—기획서 ③(M)** 점수 규칙, 실패 조건, 클리어 조건 세 가지를 순서대로 씁니다.
- **마지막—기획서 ④(A)** 수집(어항)과 감각(바다 디자인, 큰 글자)을 지정합니다.

> **내 문항을 직접 넣고 싶다면**
>
> 프롬프트 뒤에 "문제는 다음 목록을 사용해줘:"라고 쓰고 문항을 붙여넣으면, AI가 만든 문항 대신 내 문항이 들어갑니다. 우리 반 진도에 딱 맞추는 가장 확실한 방법이자, 문항 검수 부담을 절반으로 줄이는 방법이기도 합니다.

생성 과정은 3단원과 같습니다. 입력하고, Build를 누르고, 2–3분 지켜보고, 플레이합니다. 첫 결과가 기획과 다르더라도 당황할 일은 아닙니다. "어항이 안 보여. 화면 아래에 추가해줘"처럼 기획서와 다른 점부터 하나씩 맞춰 가면 됩니다.

생성 결과를 기획서 옆에 놓고 대조하기

완성된 게임을 기획서와 나란히 놓습니다. 점검 질문은 3단원에서 쓰던 다섯 가지에 하나가 더해집니다. “핵심 행동이 살아 있는가.” 학생이 이 게임에서 반복하는 행동이 정말 분수 비교인지, 아니면 클릭 반사신경인지 확인하는 것이 학습 게임 설계의 마지막 관문입니다.

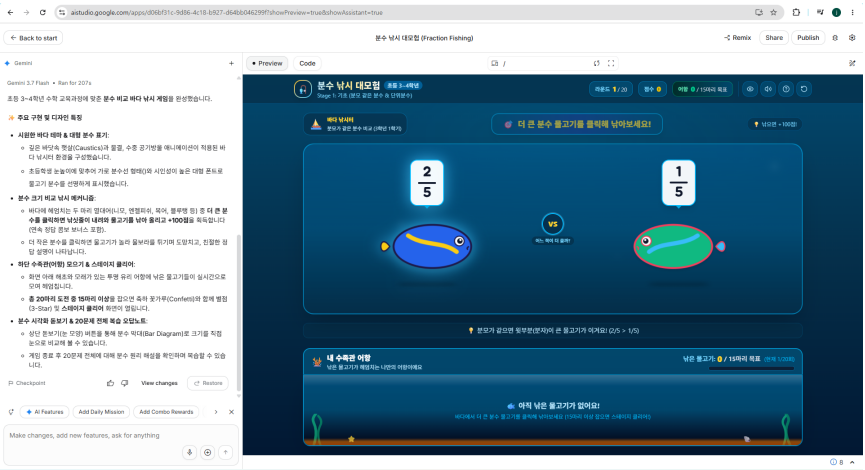


그림 4-1. 기획서대로 생성된 「분수 낚시 대모험」. 상단에 라운드·점수·어항 표시, 가운데에 비교할 두 분수, 아래에 낚은 물고기가 모이는 어항

- ☐ **학습 목표** —이 게임을 20분 하면 분수 비교가 늘겠는가.
- ☐ **핵심 행동** —학생이 반복하는 행동이 곧 학습 행동인가.
- ☐ **메커닉** —점수·실패·클리어 규칙이 기획대로 들어갔는가.
- ☐ **재미** —노린 재미인 도전·수집·감각이 실제로 느껴지는가.
- ☐ **문항** —분수 표기와 크기 비교에 오류는 없는가. 전 문항을 교사가 직접 확인했는가.

그림에서 눈여겨볼 것은 화면 위쪽 표시줄입니다. 라운드 1/20, 점수 0, 어항 0/15마리 목표가 나란히 붙어 있는데, 이 세 숫자가 곧 기획서 ③에 적은 클리어 조건이 화면으로 구현된 모습입니다. 기획서에 적어 두지 않았다면 AI는 이 표시줄을 만들 이유를 알지 못했을 것입니다.

기획서와 다른 점은 그때그때 메모해 두세요. 그 메모가 다음 절의 준비물이 됩니다.

4.5 게임 요소 강화—재미 장치 고르고 붙이기

이제 1단원에서 해부했던 보상 설계를 직접 사용합니다. 아래 여덟 요소에서 두셋만 골라 한 번에 하나씩 추가하세요. 요소마다 만들어내는 재미가 다르므로, 기획서 ④에서 노린 재미와 맞는 것을 고르는 것이 요령입니다. 많이 넣을수록 좋은 것이 아닙니다. 초점이 흐려지면 학습도 재미도 함께 잃습니다.

요소 메뉴판 여덟

요소	만드는 재미	요청 문장
타이머	긴장감·도전	“문제마다 15초 제한 시간을 추가해 줘”
콤보	몰입·성취	“연속 정답이면 콤보 보너스 점수를 넣어줘”
생명	신중함·긴장	“목숨 3개 시스템으로 바꿔줘”
레벨	성장·발견	“5문제마다 레벨이 오르고 배경이 바뀌게 해줘”
효과음·BGM	감각	“정답/오답 효과음과 배경음악을 넣어줘”
배지·보상	수집·성취	“만점이면 황금 트로피 배지를 주게 해줘”
랜덤 출제	반복 플레이	“문제 순서를 매번 섞어줘”
다시 풀기	학습 완성	“틀린 문제만 마지막에 다시 풀게 해줘”

추가하는 절차는 단순합니다. 다만 순서를 지키는 것이 중요합니다.

1. **하나만 요청합니다.** 요청 문장을 그대로 붙여넣습니다. 세 개를 한 문장에 넣으면 결과를 되돌리기 어려워집니다.
2. **바로 플레이해 봅니다.** 넣은 요소가 실제로 작동하는지, 그리고 게임의 다른 부분을 망가뜨리지 않았는지 확인합니다.
3. **기획서 ④와 대조합니다.** 노린 재미가 실제로 강해졌는지 봅니다. 아니라면 다음 요소로 넘어가지 말고 이 요소를 빼는 편이 낫습니다.

4 **그다음 하나를 요청합니다.** 두셋에서 멈춥니다.

> **보상 설계의 윤리**
> 1단원에서 해부한 보상 루프를 이제 우리가 설계하는 쪽에 섭니다. 학습을 돕는 보상(다시 풀기, 성취 배지)과 시간만 붙잡는 보상(무한 랜덤 상자)의 차이를 의식하세요. 우리의 목표는 몰입이지 중독이 아닙니다. 판단이 어렵다면 이렇게 물어보시면 됩니다. 이 보상은 학생이 무엇을 더 하게 만드는가. 문제를 더 풀게 만든다면 학습 보상이고, 화면 앞에 더 앉아 있게만 만든다면 그렇지 않습니다.

AI가 먼저 건네는 제안 칩

요소를 직접 요청하는 것 외에 다른 길도 있습니다. 채팅창 위에는 지금 게임에 어울리는 기능 제안 칩이 자동으로 나타납니다. 무엇을 요청해야 할지 떠오르지 않을 때 훑어보는 메뉴판으로 쓰기 좋습니다.

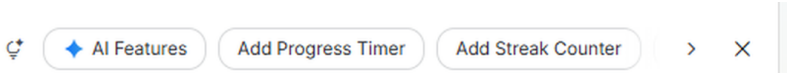


그림 4-2. 채팅창 위의 기능 제안 칩. OX 퀴즈에는 Add Progress Timer와 Add Streak Counter를 제안했다

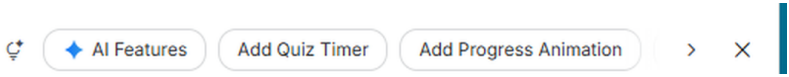


그림 4-3. 게임이 바뀌면 제안도 바뀐다. 바다 테마 게임에는 Add Quiz Timer와 Add Progress Animation을 제안했다

두 그림을 비교해 보면 제안이 게임 내용을 따라 바뀐다는 것을 알 수 있습니다. 칩을 누르면 해당 기능이 바로 추가되므로, 요청 문장을 쓸 필요조차 없습니다.

다만 칩은 어디까지나 제안입니다. 누르기 전에 기획서 ④와 맞는지 한 번 판단하세요. 맞으면 클릭 한 번으로 강화가 끝나고, 아니면 무시하면 됩니다. AI가 제안했다는 이유만으로 넣기 시작하면 요소가 다섯 개, 여섯 개로 불어나고 게임의 초점이 사라집니다.

되돌리기 안전장치—Checkpoint

과감하게 실험할 수 있는 이유는 되돌아갈 수 있기 때문입니다. 에이전트의 응답마다 Checkpoint가 자동으로 저장됩니다.

그림 4-4. 응답마다 생기는 Checkpoint 줄. View changes로 변경 내용을 보고, Restore로 그 시점으로 복원한다

새 요소를 넣었다가 게임이 이상해지면 해당 응답의 Restore 버튼을 눌러 그 시점 상태로 복원하면 됩니다. View changes를 누르면 무엇이 바뀌었는지 먼저 확인할 수도 있습니다. 요소를 하나씩 넣으라고 한 이유가 여기에도 있습니다. 하나씩 넣으면 되돌릴 지점도 하나씩 남습니다.

강화가 끝난 화면

요소 두셋이 붙으면 같은 게임이 눈에 띄게 달라집니다.

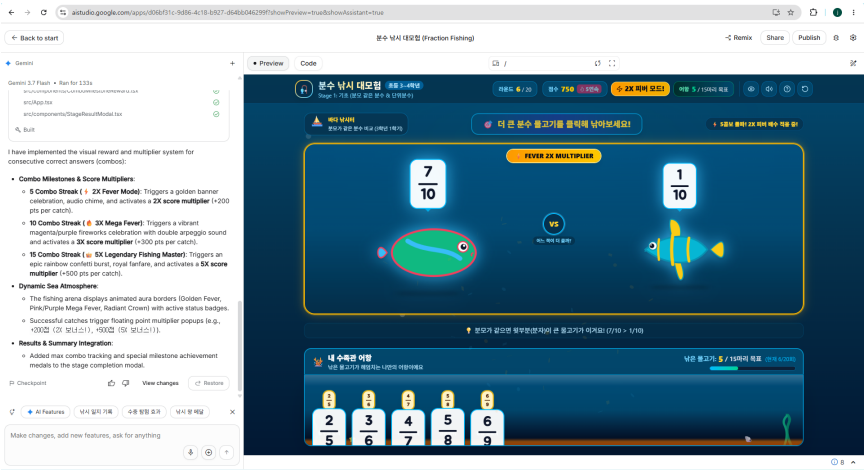


그림 4-5. 강화 결과. 콤보 5연속이 쌓이자 2X 피버 모드가 켜지고, 어항에는 낚은 분수 카드가 다섯 장 모여 있다

화면 위쪽의 점수 750 옆에 5연속 표시가 붙고, 2X 피버 모드 배지가 켜져 있습니다. 아래 어항에는 지금까지 낚은 물고기가 카드로 쌓입니다. 콤보는 몰입과 성취

를, 어항은 수집을 담당합니다. 기획서 ④에 적었던 도전·수집·감각이 각각 어느 장치로 구현되었는지 짚어 보면, 요소를 왜 그것으로 골랐는지 설명할 수 있게 됩니다. 학생에게 게임 제작을 가르칠 때도 이 설명이 그대로 수업 내용이 됩니다.

4.6 난이도 곡선·배려 설계와 AI 실시간 기능

좋은 학습 게임은 잘하는 학생만 살아남는 게임이 아니라, 모든 학생이 끝까지 도전하는 게임입니다. 난이도 곡선과 배려 장치는 게임 완성도의 마지막 조각이자, AI가 알아서 해 주지 않는 부분입니다. 교사만이 설계할 수 있습니다.

난이도 곡선—초반 성공 경험이 끝까지 가는 힘

문제 배치만 바꿔도 게임의 성격이 달라집니다. "문제를 쉬움 5개 → 보통 3개 → 어려움 2개 순서로 배치해줘"라고 요청하세요. 처음 몇 문제에서 맞히는 경험을 한 학생은 뒤의 어려운 문제에서 한 번 틀려도 계속합니다. 반대로 첫 문제부터 막힌 학생은 세 번째 문제쯤에서 손을 놓습니다. 1단원에서 본 카이와의 아곤, 즉 경쟁의 재미는 이길 가능성이 있을 때만 성립합니다.

난이도 선택권과 틀린 문제 재도전

"시작할 때 쉬움/보통/어려움 모드를 고르게 해줘"는 같은 게임을 수준별 과제로 바꾸는 가장 간단한 방법입니다. 별도의 학습지를 세 종류 만들 필요 없이, 학생이 스스로 고르게 하면 됩니다.

"틀린 문제만 마지막에 다시 풀게 하고, 두 번째에 맞히면 절반 점수를 줘"는 실패를 학습 기회로 되돌리는 장치입니다. 점수를 절반만 주는 이유는 처음부터 맞힌 학생의 성취를 지키면서도 다시 풀 이유를 만들기 위해서입니다. 요소 메뉴판의 다시 풀기와 같은 장치인데, 학습 완성이라는 목적에서 보면 이쪽이 본래 자리입니다.

읽기·색·대비 배려

읽기 속도가 게임 실력이 되어서는 안 됩니다. "문제를 소리로 읽어주는 버튼을 넣어줘"라고 요청하면 음성 읽기(TTS) 기능이 붙고, "글자를 더 크게, 문장을 짧게"는 저학년 게임에서 거의 언제나 필요한 요청입니다.

색도 마찬가지입니다. "빨강-초록 대신 파랑-주황으로 구분하고, 모양 차이도 함께 줘"라고 요청하세요. 색각 이상이 있는 학생도 구분할 수 있는 신호 설계이며, 색 하나에만 의존하지 않는 화면은 교실 TV의 색이 바래 있을 때도 잘 읽힙니다.

> **배려 설계는 특별한 학생만을 위한 것이 아닙니다**
>
> 음성 읽기는 읽기가 느린 학생에게 필요하지만, 소란한 교실에서 모두에게 도움이 됩니다. 큰 글자는 저학년에게 필요하지만, 뒷자리에서 TV를 보는 학생에게도 필요합니다. 배려 설계는 언제나 모든 학생의 경험을 좋게 만듭니다. 보편적 학습 설계(UDL)가 말하는 바가 이것입니다.[a]
>
> [a] CAST. (2018). *Universal Design for Learning Guidelines version 2.2.* https://udlguidelines.cast.org

AI 실시간 기능—문제가 무한히 샘솟는 게임

지금까지 만든 게임은 만들 때 정해진 문항을 보여 주는 정적인 게임이었습니다. AI Studio로 만든 앱에는 Gemini를 호출하는 기능을 넣을 수 있어서, 게임이 플레이 중에 실시간으로 문제를 만들거나 학생의 답을 채점하게 할 수 있습니다. 정해진 문항과는 차원이 다른 가능성이 열립니다.

- **무한 문제 은행** —"매 판마다 Gemini가 새로운 분수 비교 문제를 만들어 내게 해줘". 문제가 소진되지 않아 반복 플레이가 가능해집니다.
- **서술형 채점 코치** —"학생이 문장으로 답을 쓰면 Gemini가 채점하고 한 줄 조언을 주게 해줘". 객관식을 넘어 서술형 게임이 됩니다.
- **스무고개 AI 술래** —"Gemini가 과학 개념 하나를 마음속에 정하고, 학생의 예/아니오 질문에 답하는 스무고개 게임을 만들어줘".
- **끝말잇기 심판** —"학생과 Gemini가 번갈아 낱말을 잇고, Gemini가 규칙 위반을 판정하게 해줘". 국어 어휘 수업의 단골 손님입니다.

기능이 켜졌는지 확인하는 법

1. **두세 판을 연속으로 플레이합니다.** 문제가 매번 달라지는지 확인합니다. 같은 문제가 반복되면 아직 정적 문항을 쓰고 있는 것입니다.
2. **로딩을 정상으로 받아들입니다.** 문제가 나올 때 잠깐의 지연이 생기는 것은 Gemini를 호출하는 정상 동작입니다.
3. **범위를 좁혀 다시 요청합니다.** 수준을 벗어난 문제가 나오면 "3–4학년 교육과정 범위 안에서만"이라고 범위를 명시해 다시 요청합니다.

> **실시간 생성 문항은 미리 검수할 수 없습니다**
> AI가 플레이 중에 만드는 문제는 교사가 사전에 확인할 방법이 없습니다. 학년 수준과 소재 범위를 프롬프트에 분명히 제한하고("3학년 수준, 교과서 어휘만"), 수업 전에 반드시 여러 판을 시험 플레이하세요. 미리보기 화면에서는 바로 동작하며, 외부에 배포할 때의 설정은 5단원에서 다룹니다.

4.7 플레이테스트—두 개의 시점으로 검증하기

게임 개발의 마지막 단계는 언제나 플레이테스트입니다. 교사 시점과 학생 시점, 두 렌즈로 점검한 뒤 동료와 교차 테스트합니다. MDA의 언어로 말하면, 설계자가 만든 규칙(M)이 의도한 재미(A)로 도착했는지를 플레이어의 눈으로 확인하는 과정입니다.

교사 시점 점검—학습이 일어나는가

- □ 문항의 사실 오류·표기 오류가 없는가. 전 문항을 직접 확인했는가.
- □ 핵심 행동이 학습 행동과 일치하는가(79쪽의 ②번 칸).
- □ 한 판의 길이가 수업 흐름에 맞는가. 5–10분을 권합니다.
- □ 가장 느린 학생도 끝까지 갈 수 있는가. 배려 설계를 확인합니다.

학생 시점 점검—재미가 도착했는가

- □ 첫 30초 안에 무엇을 해야 하는지 알 수 있는가.
- □ 실패했을 때 "다시 하고 싶은가", 아니면 "그만두고 싶은가".
- □ 점수·보상이 오를 때 기분이 좋은가. 효과가 체감되는가.
- □ 노린 재미 두셋이 실제로 느껴지는가.

첫 항목이 가장 자주 걸립니다. 만든 사람은 규칙을 알고 있어서 30초 테스트를 스스로 할 수 없습니다. 그래서 다음 절차가 필요합니다.

동료 교차 테스트 프로토콜—10분

연수 현장이라면 옆자리와, 학교라면 동학년 선생님과 다음 순서로 진행합니다.

1. **게임을 바꿔 3분씩 플레이합니다.** 만든 사람은 설명하지 않습니다. 설명하지 않는다는 규칙이 곧 첫 30초 테스트를 겸합니다.
2. **칭찬 2개와 제안 1개를 교환합니다.** 칭찬을 먼저 두 개 말하게 하는 것은 예의 때문이 아니라, 무엇이 이미 작동하고 있는지 설계자가 알아야 하기 때문입니다.
3. **받은 제안 중 하나를 즉시 반영합니다.** 그 자리에서 대화로 수정합니다. 수정 전 상태는 Checkpoint가 지키고 있으니 과감하게 반영해 보세요(86쪽).

이 루틴은 그대로 학생 게임 제작 수업의 상호 평가 활동으로 옮겨 쓸 수 있습니다. 초등학생에게도 "칭찬 두 개, 제안 하나"는 그대로 통합니다.

4.8 60분 미션—내 수업의 본격 게임을 완성하라

기획서부터 플레이테스트까지 전 과정을 자신의 교과로 한 바퀴 돌립니다. 3단원 미션보다 깁니다. 결과물은 6단원까지 계속 발전시킬 "나의 대표 게임"이 되므로, 소재는 실제로 다음 달에 가르칠 단원에서 고르시기 바랍니다.

0–10분 **미니 기획서 작성**

성취기준 한 줄에서 출발해 핵심 행동, 메커닉, 노리는 재미 두셋까지 종이에 먼저 씁니다(79쪽).

10–25분 **장르 선택과 생성**
카탈로그에서 장르를 고르고, 기획서를 프롬프트로 옮겨 New app에서 생성합니다(81쪽).

25–40분 **게임 요소 강화**
메뉴판과 제안 칩에서 두셋을 골라 하나씩 추가합니다. 이상해지면 Restore로 되돌립니다(86쪽).

40–50분 **난이도·배려 설계**
난이도 곡선 또는 모드 선택에 배려 장치 하나 이상을 더합니다(89쪽).

50–60분 **플레이테스트**
두 시점으로 점검한 뒤 동료 교차 테스트를 합니다. 칭찬 2, 제안 1, 즉시 수정 1(91쪽).

완성 기준

여섯 칸이 모두 채워지면 이 단원의 미션은 끝입니다.

- ☐ 내 교과 내용이 담겨 있다.
- ☐ 게임 요소 두 개 이상이 실제로 작동한다.
- ☐ 배려 장치가 하나 이상 있다.
- ☐ 전 문항을 교사가 검수했다.
- ☐ 동료 한 명이 플레이해 보았다.
- ☐ 「과목_주제_장르」 형식으로 이름을 붙여 저장했다.

미션 기록장

게임 이름: _____

성취기준: _____

고른 장르: _____ 핵심 행동: _____

넣은 요소: _____ · _____ · _____

배려 장치: _____

동료가 준 제안 1개: _____

> 전 문항 검수: ☐ 동료 플레이: ☐ 이름 저장: ☐

이 게임은 5단원에서 학생들에게 배포되고, 6단원에서 실시간 리더보드가 붙습니다. 오늘의 완성도가 남은 여정의 출발선입니다.

4.9 교실 적용—언제, 어떻게 쓰는 게임인가

게임이 완성되어도 "수업 어디에 넣을 것인가"는 별도의 설계입니다. 같은 게임도 배치에 따라 역할이 달라집니다. 아래 세 가지 배치와 두 가지 운영 방식을 조합하면 대부분의 수업 상황을 덮을 수 있습니다.

도입 5분—동기 유발

전 차시 복습을 겸해 오늘 주제를 맛보게 합니다. 쉬움 모드로 시작해 전원에게 성공 경험을 주고 넘어가는 것이 요령입니다. 5분을 넘기지 않습니다. 길어지면 본 수업이 밀립니다.

정리 10분—형성평가

오늘 배운 내용으로 한 판 돌립니다. 결과 화면의 점수와 오답이 곧 형성평가 데이터가 됩니다. 교사는 순회하며 어느 유형에서 많이 막히는지만 보아 두면, 다음 차시 도입에서 무엇을 다시 짚을지가 정해집니다.

단원 마무리—복습 대회

단원 전체 문항으로 확장해 모둠 대항전을 엽니다. 6단원에서 붙일 리더보드가 가장 빛나는 자리이기도 합니다. 다만 순위를 성적이나 상점에 연결하지 않는다는 원칙은 여기서도 그대로입니다.

운영 방식 두 가지

- **전체 화면·교사 기기 한 대** —TV나 빔에 게임을 띄우고 모둠별로 번갈아 답합니다. 학생 기기가 없어도 되고 전체 분위기가 뜨겁습니다. 골든벨식 진행에 적합하며, 프롬프트에서 글자 크기를 미리 크게 지정해 둔 이유가 여기에 있습니다.
- **개별 기기·태블릿이나 노트북** —학생마다 자기 속도로 플레이합니다. 개별 연습과 수준별 과제에 적합하며, 링크 배포가 필요합니다. 바로 다음 5단원에서 URL과 QR로 배포하는 방법을 실습합니다.

> **지도안 한 줄 틀**
>
> 〔단원〕의 〔성취기준〕 도달을 위해 〔게임명〕을 〔배치〕에 〔운영 방식〕으로 활용하여 〔핵심 행동〕을 반복 연습한다.

4.10 자주 막히는 지점 여섯

요소를 추가하는 단계에서는 3단원과 다른 유형의 문제가 나타납니다. 공통 해법은 두 가지입니다. 하나씩 요청하기, 그리고 Checkpoint의 Restore.

Q. 기능을 추가했더니 게임이 망가졌어요

A. 해당 요청 이전 응답의 Restore 버튼을 누르면 그 시점으로 복원됩니다(86쪽). 그다음 요청을 더 작게 쪼개서 다시 시도하세요.

Q. 여러 기능을 한꺼번에 요청했더니 이상해요

A. "타이머와 콤보와 레벨을 넣어줘"보다 세 번에 나눠 요청하세요. 하나 넣고 플레이로 확인하고, 또 하나 넣는 순서가 결국 가장 빠릅니다.

Q. 그림이나 캐릭터가 어색해요

A. "캐릭터를 이모지로 바꿔줘" 또는 "도형과 색으로만 단순하게 표현해줘"가 안정적입니다. 저작권 걱정도 함께 사라집니다.

Q. 게임이 무겁고 느려져요

A. "애니메이션 효과를 줄여서 가볍게 만들어줘"라고 요청하세요. 교실의 저사양 기기일수록 단순한 게임이 좋습니다.

Q. 수정을 반복했더니 대화가 너무 길어요

A. 완성 시점마다 이름을 붙여 저장해 두고, 큰 방향 전환은 New app에서 새로 시작하는 편이 결과가 깨끗합니다.

Q. 아이디어가 안 떠올라요

A. 채팅창 위 제안 칩을 훑어보거나, Gallery에서 다른 앱을 리믹스해 구조만 빌려오세요. 기획서 ④번 칸으로 돌아가 노린 재미를 다시 읽어 보는 것도 좋습니다(79쪽).

4.11 단원 마무리

이제 선생님에게는 퀴즈가 아니라 게임이 하나 있습니다. 성취기준에서 출발해 규칙으로 번역되고, 재미 장치가 붙고, 모든 학생이 끝까지 갈 수 있게 다듬어진 게임입니다.

이 단원에서 우리는 프롬프트보다 기획서가 먼저라는 것을 확인했습니다. 성취기준에서 핵심 행동으로, 핵심 행동에서 메커닉으로, 메커닉에서 노리는 재미로 이어지는 네 칸을 채우고 그것을 문장으로 옮겨 게임을 생성했습니다. 장르는 기술이 아니라 교과 특성이 고른다는 기준을 세웠고, 반복 연습이면 수집형과 디펜스, 개념 관계면 카드 배틀, 탐구 과정이면 방탈출이라는 지도를 손에 넣었습니다.

이어서 요소 메뉴판과 제안 칩으로 재미를 강화했습니다. 두셋만, 하나씩, 그리고 언제든 Restore로 되돌릴 수 있다는 원칙이 실험을 겁내지 않게 해 주었습니다. 몰입은 설계하되 중독은 설계하지 않는다는 기준도 함께 세웠습니다. 마지막으로 난이도 곡선과 배려 장치, AI 실시간 기능을 더하고 두 시점의 플레이테스트로 검증했습니다. 잘하는 학생만 살아남는 게임이 아니라 모두가 끝까지 하는 게임, 그것이 교사가 만드는 학습 게임의 차별점입니다.

남은 숙제. 완성한 게임을 다음 주 수업 한 차시에 실제로 넣어 보고, 학생 반응과 걸린 시간을 한 문장으로 기록해 두세요. 특히 가장 느린 학생이 몇 문제까지 갔는지를 적어 두면 다음 수정의 기준이 됩니다.

다음 단원 예고. WORLD 5 ·온라인 출판—내 게임에 인터넷 주소를 달아 줍니다. 링크 한 줄이면 교실 태블릿에서도 학생의 집 스마트폰에서도 열리게 만드는 것이 다음 단원의 목표입니다.

기획서 한 장이 있으면 게임은 다시 만들 수 있습니다. 이 단원이 남기는 것은 게임 한 편이 아니라 그 한 장을 쓰는 습관입니다.

참고문헌

Hunicke, R., LeBlanc, M., & Zubek, R. (2004). *MDA: A formal approach to game design and game research.* AAAI Workshop.

Prensky, M. (2001). *Digital game-based learning.* McGraw-Hill.

Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011). *From game design elements to gamefulness.* MindTrek 2011.

CAST. (2018). *Universal Design for Learning Guidelines version 2.2.* https://udlguidelines.cast.org

Google. (2026). *Build apps in Google AI Studio.* Gemini API 개발자 문서. https://ai.google.dev/gemini-api/docs/aistudio-build-mode

화면 캡처: Google AI Studio, 2026년 8월. 서비스 정책에 따라 화면 구성은 변경될 수 있습니다.

WORLD 5

온라인 출판

내 게임에 주소를 달아주기

4단원까지 우리는 “내 컴퓨터 화면 안에서만 도는 게임”을 만들었습니다. 이 단원의 목표는 단 하나입니다. 그 게임에 인터넷 주소를 붙여, 학생이 교실 태블릿이든 집 스마트폰이든 링크 하나로 열 수 있게 만드는 것.

설치도, 파일 전달도, 로그인도 필요 없습니다. 주소만 있으면 됩니다.

이 단원은 교사에게 낯선 도구인 GitHub를 정면으로 다룹니다. 그래서 설명을 줄이고 화면을 늘렸습니다. 본문의 그림은 2026년 8월에 실제 GitHub와 Google AI Studio에서 촬영한 것이며, 개인정보에 해당하는 이메일은 흐리게 처리했습니다. 서비스 정책에 따라 버튼의 위치나 이름이 바뀔 수 있으니, 그림과 다르면 같은 뜻의 메뉴를 찾으시면 됩니다.

이 단원의 도착점은 다음 한 줄입니다.

> **이 단원의 도착점**
>
> https://jjangsamnet.github.io/adventure_math/
>
> 4단원에서 만든「분수 낚시 대모험」이 이 주소로 열립니다. 실습을 마치면 선생님도 자기 이름이 들어간 주소를 하나 갖게 됩니다.

5.1 왜“파일”이 아니라“주소”여야 할까

많은 선생님이 자료를 만들면 파일로 나눠줍니다. 한글 파일이든 PDF든, 만든 것을 그대로 건네주는 것이 가장 자연스러운 방식이기 때문입니다. 그러나 학습 게임은 파일로 배포하는 순간 절반이 죽습니다.

이유는 세 가지입니다. 첫째, 기기마다 열리는 앱이 다릅니다. 같은 HTML 파일이 어느 태블릿에서는 잘 열리고 어느 스마트폰에서는 코드가 그대로 보입니다. 모바일에서는 압축을 풀 수 없어 아예 시작조차 못 하는 경우도 흔합니다. 둘째, 고칠 때마다 다시 나눠줘야 합니다. 문항 하나를 바꿔도 서른 명 전원에게 새 파일을 보내야 하고, 누가 옛 버전을 쓰고 있는지 알 수 없습니다. 셋째, 한 번 나간 파일은 회수할 수 없습니다.

주소는 이 세 문제를 한 번에 해결합니다.

구분	파일로 나눠주기	주소로 열게 하기
학생 기기	기기·브라우저마다 다르게 열림. 모바일에서 압축 해제 불가	주소창에 붙여넣으면 어디서나 동일하게 열림
수정 반영	고칠 때마다 전원에게 다시 배포	한 번 고치면 모두가 즉시 최신 버전
가정 연계	가정 기기로 옮기기 번거로움	문자·클래스룸에 링크 한 줄, QR 한 장
회수·중단	이미 받은 파일은 회수 불가	게시를 끄면 즉시 닫힘
교사 부담	기기 문제 문의가 교사에게 집중	“주소 다시 알려주세요” 수준으로 감소

> **교실 관찰**
> 링크 배포로 바꾼 뒤 가장 크게 줄어드는 것은 “안 열려요”라는 말입니다. 수업 도입 5분이 통째로 돌아옵니다.

5.2 출판 방법 네 가지

AI Studio로 만든 앱을 세상에 내보내는 길은 크게 네 가지입니다. 네 가지는 우열이 아니라 용도가 다릅니다. 특히 “AI 기능이 살아 있는가”와 “주소가 얼마나 오래 가는가” 두 가지를 눈여겨보세요. 이 둘이 교실에서의 선택을 결정합니다.

첫째, Share 링크—동료와 나누기

AI Studio 안에서 다른 사용자에게 앱을 열어줍니다. 무료지만 공유받은 사람이 코드를 볼 수 있고, 그 앱에서 발생하는 AI 호출이 내 사용량으로 차감됩니다. 동료 교사나 연수 참가자와 나누기에는 최적이지만, 학생 서른 명에게 각각 권한을 주기에는 번거롭습니다. 주소는 AI Studio 계정에 종속됩니다.

둘째, Publish —Cloud Run에 게시하기

버튼 세 번으로 실제 인터넷 주소가 발급됩니다. AI 호출에 쓰는 열쇠(API 키)가 서버 쪽에 보관되므로 학생에게 노출되지 않습니다. Starter Tier 덕분에 Google Cloud 프로젝트를 따로 만들거나 결제 계정을 등록하지 않아도 앱 두 개까지 게시할 수 있습니다. AI가 실시간으로 작동하는 게임을 학생에게 배포할 때의 정답입니다.

셋째, GitHub + Pages —무료·영구 주소

코드를 내 저장소에 올리고 정적 사이트로 공개합니다. 무료이고 기한이 없으며 내 계정 소유입니다. 다만 서버가 없으므로 실행 중 AI를 호출하는 기능은 작동하지 않습니다. 코드만으로 완결되는 게임, 여러 해에 걸쳐 쓸 학급 자료에 어울립니다. 이 책의 실습은 이 길로 갑니다.

넷째, 내려받아 외부 호스팅에 올리기

코드를 압축 파일로 내려받아 학교 홈페이지나 다른 호스팅에 올립니다. 학교 서버 정책상 외부 서비스가 막혀 있을 때 쓰는 우회로입니다. 비용과 수명은 올린 곳의 정책을 따릅니다.

5.3 갈림길—딱 하나만 물어보세요

네 가지를 다 외울 필요는 없습니다. 질문 하나면 길이 갈립니다. "내 게임은 실행 중에 AI에게 말을 거는가?" AI에게 말을 건다는 것은 코드가 Gemini API를 호출한다는 뜻이고, 그 호출에는 열쇠가 필요합니다. 정적 호스팅인 GitHub Pages에는 이 열쇠를 안전하게 숨길 곳이 없습니다.

예—AI가 실시간으로 작동한다면

학생 답을 AI가 채점하고 피드백을 주거나, 매번 새 문제를 만들어 주거나, 음성으로 문항을 읽어 주거나, AI 캐릭터와 대화하는 게임이라면 여기에 해당합니다. 이

런 게임은 **Publish(Cloud Run)**로 갑니다. API 키가 서버 쪽에 남아 학생에게 보이지 않습니다.

아니오—코드만으로 완결된다면

문항과 규칙이 코드 안에 이미 들어 있는 게임—4단원의 분수 낚시 대모험, OX 퀴즈, 타워 디펜스 같은 것들—은 실행 중에 AI를 부르지 않습니다. 호출할 AI가 없으니 숨길 열쇠도 없습니다. **GitHub Pages**로 가면 무료로, 기한 없이, 내 계정 소유의 주소를 갖게 됩니다.

> **API 키 경고**
>
> AI 호출 코드를 그대로 GitHub 공개 저장소에 올리면 키가 코드 안에서 그대로 읽힙니다. 남이 내 키로 요금을 쓰게 됩니다. 공개 저장소의 키는 자동 수집 대상이기도 합니다. AI 기능이 있는 앱은 반드시 Publish 쪽으로 보내세요.

5.4 공유 링크 만들기

앱을 만들면 화면 위쪽 오른편에 버튼 세 개가 나타납니다. 이름이 비슷해 헷갈리기 쉬운데, 하는 일은 완전히 다릅니다. 세 버튼의 차이를 먼저 정리하고 시작합니다.

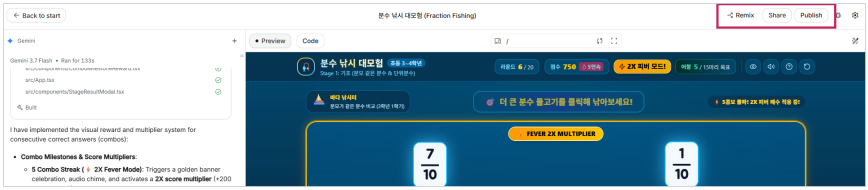


그림 5-1. 앱 상단 툴바. 오른쪽 끝에 Remix · Share · Publish 세 버튼이 있다

Remix는 이 앱을 복제해 내 것으로 만듭니다. 원본은 두고 변형할 때 쓰며, 출판과는 무관합니다. Share는 AI Studio 안에서 다른 사용자에게 앱을 열어줍니다. 상대는 실행 화면과 코드를 함께 봅니다. Publish는 실제 인터넷 주소로 게시합니다.

버튼이 보이지 않는다면 3단원에서 배운 Preview / Code 탭 중 Preview 쪽인지 확인하세요. 오른쪽 위 톱니바퀴를 누르면 같은 기능이 Chat / Share / Publish / Versions / GitHub / Integrations / Secrets 탭으로도 열립니다. 이 단원에서는 이 중 Share ·Publish ·GitHub 세 탭만 씁니다.

Share 대화상자 다루기

앱은 기본이 비공개입니다. Share는 그중 가장 가벼운 방법으로, 지정한 사람에게만 앱을 열어줍니다.

1. Share를 클릭하면 대화상자가 열립니다.
2. General access를 확인합니다. 기본은 "Restricted: Only people you specify can access" —지정한 사람만 접근입니다.
3. 이메일을 입력해 참여자를 추가합니다. 소유자(Owner)는 나입니다.
4. Settings의 두 토글을 봅니다. Default to fullscreen은 전체화면으로 열기, Include your Gemini chat history는 내 대화 기록을 함께 공개할지입니다. 두 번째는 꺼 두는 편이 안전합니다.
5. 맨 아래 Copy link로 링크를 복사해 전달합니다.

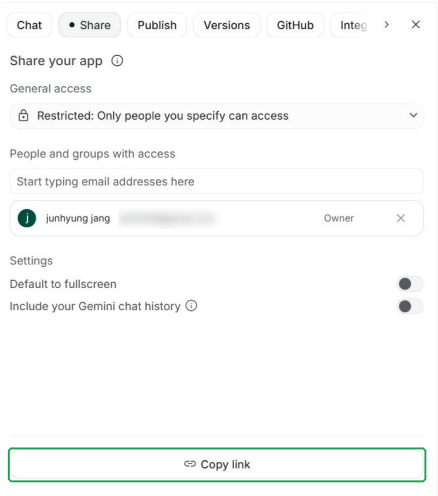


그림 5-2. Share 대화상자. 공개 범위와 참여자, 맨 아래 Copy link(초록 표시)

학생 배포에는 권하지 않습니다

첫째, 상대가 코드를 볼 수 있습니다. 둘째, 공유한 앱의 API 호출이 내 사용량으로 차감됩니다. 셋째, 서른 명에게 각각 권한을 주는 일이 번거롭습니다. 동료 교사·연수 참가자와 나누는 용도로 쓰세요.

5.5 Cloud Run에 게시하기

Publish는 앱을 Google Cloud Run에 올려 실제 서비스처럼 돌립니다.[1] AI 호출에 쓰는 API 키가 서버 쪽에 보관되므로 학생에게 노출되지 않습니다. Starter Tier 덕분에 Google Cloud 프로젝트를 따로 만들거나 결제 계정을 등록하지 않아도 앱 두 개까지 게시할 수 있습니다.

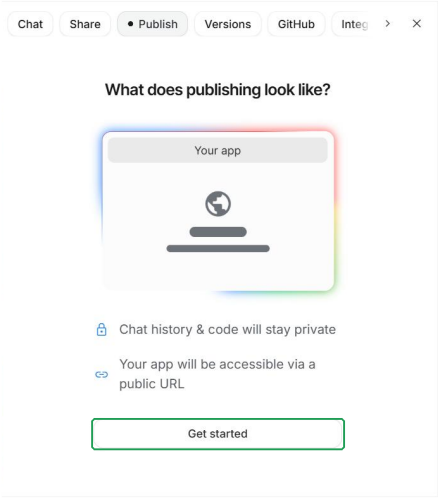


그림 5-3. Publish 패널. Get started(초록 표시)를 누르면 게시가 시작된다

[1] Google. (2026). *Deploying from Google AI Studio.* Gemini API 개발자 문서. https://ai.google.dev/gemini-api/docs/aistudio-deploying

세 번 누르면 끝납니다

Publish → Get started → Publish App. 배포가 끝나면 접속용 Cloud Run 주소가 발급됩니다. 패널에 적힌 두 줄—"Chat history & code will stay private"과 "Your app will be accessible via a public URL" —은 대화 기록과 코드는 비공개이고 앱만 공개 주소로 열린다는 뜻입니다. ai.studio 하위 도메인을 붙여 조금 더 짧은 주소를 만들 수도 있는데, 전 세계에서 유일해야 하며 선착순으로 배정됩니다.

Starter Tier의 한도는 앱 두 개, 단일 리전 배포입니다. 이미 Google Cloud 결제 계정을 쓰고 있는 사용자는 이 무료 등급의 대상이 아닙니다.

주소를 받은 다음—챙길 것 다섯

Publish가 끝나면 주소가 나옵니다. 여기서 끝내지 말고 다음 다섯 가지를 확인해야 수업 당일 사고가 없습니다. 특히 첫째와 다섯째는 실제 연수에서 가장 자주 문제가 된 항목입니다.

1. **교실 기기로 접속 테스트.** 내 노트북에서 되는 것과 교실 태블릿에서 되는 것은 다릅니다. 학교 망 필터링에 Cloud Run 도메인이 걸리는 사례가 있습니다. 수업 전날 반드시 한 대로 확인하세요.
2. **첫 로딩 시간 확인.** Cloud Run은 접속이 없으면 잠들었다가 첫 요청에서 깨어납니다. 첫 학생이 몇 초 더 기다릴 수 있습니다. 수업 시작 전 교사가 먼저 한 번 열어 두면 해결됩니다.
3. **앱 두 개 한도 관리.** 세 번째를 올리려면 기존 게시를 내려야 합니다. 학기 중 여러 게임을 돌린다면 미리 계획하세요.
4. **주소를 기록해 둘 것.** 주소는 길고 기억하기 어렵습니다. 학급 자료 폴더나 수업 지도안에 바로 적어 두세요. 다음 학년도에 다시 찾을 때 고생합니다.
5. **AI 기능이 실제로 도는지.** 배포본에서 음성·자동 채점 같은 기능을 직접 눌러 보세요. 개발 화면에서는 되고 배포본에서는 안 되는 경우가 있습니다.

> AI 기능이 없는 게임이라면 여기서 멈추지 마세요. 다음 절부터 이어지는 GitHub 길이 더 오래 가는 무료 주소를 줍니다.

5.6 GitHub 기초—교사에게 필요한 개념 넷

GitHub는 프로그래머의 세계처럼 보이지만, 교사에게 필요한 개념은 네 개뿐입니다. 아래 비유는 교실에서 학생에게 설명할 때도 그대로 쓸 수 있습니다. 6학년 학생도 "학급 문집 보관함"이라고 하면 이해합니다.

저장소(Repository) —프로젝트가 통째로 들어가는 폴더

학급 문집을 보관하는 서랍 한 칸입니다. 게임 하나에 서랍 하나를 씁니다. 이번 실습의 서랍 이름은 adventure_math입니다.

커밋(Commit) —"여기까지 저장" 하고 도장 찍기

문집을 고칠 때마다 "3월 12일, 표지 색 바꿈"이라고 적어 두는 기록장입니다. 언제든 그 시점으로 되돌아갈 수 있습니다.

푸시와 풀(Push / Pull) —올리기와 내려받기

Push는 내 서랍의 내용을 인터넷 서랍에 올리는 일, Pull은 인터넷 서랍의 최신본을 내 쪽으로 가져오는 일입니다. 이 단원에서는 Push만 씁니다.

공개와 비공개(Public / Private) —누구나 볼 수 있는지

무료 계정에서 Pages로 주소를 만들려면 저장소가 공개여야 합니다.[2] 그래서 학생 개인정보가 코드에 절대 들어가면 안 됩니다. 이 원칙은 127쪽의 점검표로 다시 돌아옵니다.

> **계정 만들기**
>
> github.com → Sign up → 이메일·비밀번호·사용자명 입력 → 이메일 인증. 사용자명은 곧 내 주소의 일부(https://사용자명.github.io/…)가 되니 교사용 아이디를

[2] GitHub. (2026). *What is GitHub Pages? / GitHub Pages limits.* GitHub Docs. https://docs.github.com/pages

> 권합니다. 가입 직후 2단계 인증(2FA) 화면이 뜨면 안내대로 설정하면 되고, 무료 계정으로 이 단원의 모든 실습이 가능합니다. 연수에서는 계정 개설을 사전 과제로 돌리면 15분을 아낄 수 있습니다.

5.7 GitHub 화면 익히기

개념을 알았으니 화면을 봅니다. 교사가 실제로 쓰는 화면은 셋뿐입니다. 로그인 후 첫 화면, 내 저장소 목록, 그리고 새 저장소를 만드는 폼. 이 셋만 눈에 익혀 두면 길을 잃지 않습니다.

로그인 후 첫 화면

github.com에 로그인하면 다음 화면이 나옵니다. 처음 보면 정보가 많아 당황스럽지만, 교사가 쓰는 곳은 두 군데뿐입니다. 왼쪽 위의 초록색 New 버튼과 그 아래 내 저장소 목록입니다.

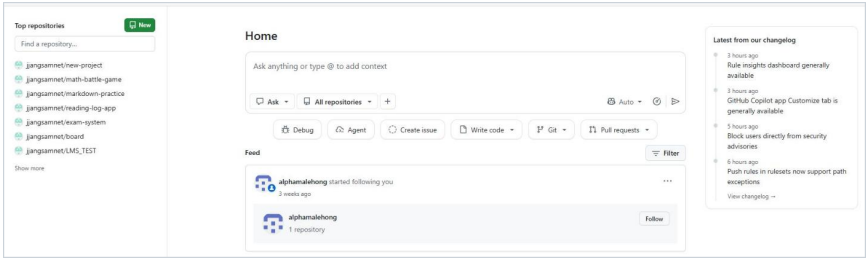


그림 5-4. GitHub 로그인 후 첫 화면. 왼쪽 위 New 버튼과 Top repositories

가운데 영역은 내가 팔로우하는 사람과 인기 저장소 소식입니다. GitHub가 원래 프로그래머들이 서로 코드를 보여주고 함께 고치는 곳이라 첫 화면이 소셜 미디어처럼 생겼습니다. 교사는 그 기능을 쓸 일이 없으니 가운데는 무시하고, 왼쪽 New와 저장소 목록만 보면 됩니다. 오른쪽 위 프로필을 눌러 Your repositories로 가면 다음 화면입니다.

내 저장소 목록—서랍장 전체 보기

프로필의 Repositories 탭입니다. 내가 가진 서랍이 전부 여기 모입니다. 각 줄에 저장소 이름, 공개 범위 배지, 설명, 사용 언어, 마지막 수정 시각이 표시됩니다.

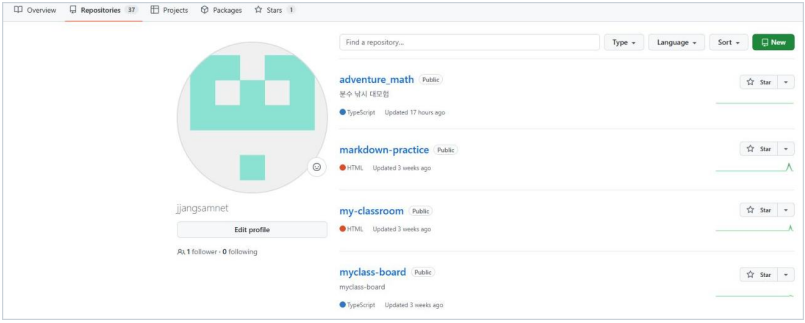


그림 5-5. 프로필 → Repositories. 저장소 이름·공개 배지·설명·언어·수정 시각

공개 배지를 먼저 봅니다. Public이면 전 세계가 코드를 볼 수 있습니다. 무료 계정으로 Pages 주소를 만들려면 Public이어야 하므로, 학생 개인정보가 없는지가 늘 전제입니다. 설명은 꼭 적어 두세요. "분수 낚시 대모험"처럼 한글 설명을 넣으면 1년 뒤에도 무엇인지 알아봅니다. 저장소 이름은 영문이라 헷갈립니다. 같은 게임을 여러 번 만들었다면 "Updated "를 보고 최신 것을 고릅니다.

새 저장소 만들기—각 칸이 무슨 뜻인가

New 버튼을 누르면 다음 화면입니다. 칸이 많아 보이지만 교사가 정할 것은 두 개뿐입니다. 이름과 공개 범위. 나머지는 기본값 그대로 두면 됩니다.

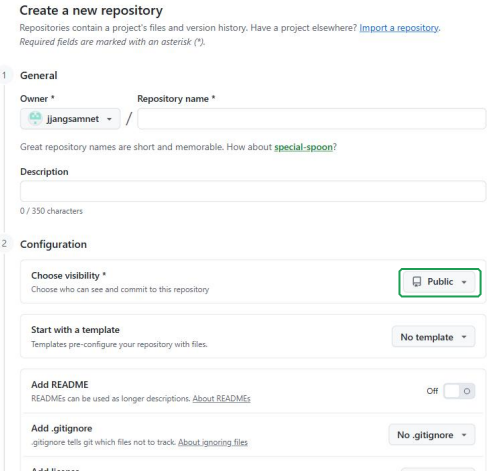


그림 5-6. Create a new repository(github.com/new). 초록 표시가 공개 범위

- **Owner / Repository name** —소유자(내 계정)와 저장소 이름. 영문 소문자·하이픈·밑줄을 씁니다. 이 이름이 주소의 뒷부분이 됩니다.
- **Description** —한글 설명. 선택 사항이지만 꼭 적으세요.
- **Choose visibility** —Public(누구나 열람) / Private(나만). 무료 계정으로 Pages를 켜려면 Public이어야 합니다.
- **Start with a template** —기존 틀에서 시작하기. 쓰지 않습니다.
- **Add README** —설명 문서를 함께 만들지 여부. AI Studio가 코드를 올릴 것이므로 꺼 두어도 됩니다.
- **Add .gitignore / Add license** —올리지 않을 파일 목록과 저작권 이용 조건. 지금은 비워 둡니다.
- **Create repository** —초록 버튼을 누르면 빈 저장소가 만들어집니다.

5.8 저장소는 누가 만드는가—두 갈래

저장소는 두 가지 방법으로 만들 수 있습니다. 결과물은 같지만 실습 흐름이 다릅니다. 연수에서는 둘째 길을 권합니다. 화면을 오가지 않아 막히는 사람이 적습니다.

첫째 길·내가 직접 만들기

github.com/new에서 저장소를 먼저 만들고, AI Studio에서 그 저장소를 골라 연결합니다. 무엇이 만들어지는지 눈으로 보고 이름·설명·공개 범위를 내가 정한다는 장점이 있지만, 화면을 두 번 오가야 해서 초보자는 헷갈릴 수 있습니다. 이미 있는 저장소에 이어 올릴 때는 이 길뿐입니다.

둘째 길·AI Studio가 만들게 하기

AI Studio의 GitHub 탭에서 이름과 공개 범위를 정하면 저장소 생성과 연결이 한 번에 끝납니다. 화면을 옮기지 않아 실습이 매끄럽고 연수 시간이 짧을 때 유리합니다. 앞 절에서 첫째 길의 화면을 먼저 본 이유는, 둘째 길이 무엇을 대신 해 주는지 알아야 문제가 생겼을 때 손댈 수 있기 때문입니다. 다음 절부터 이 길로 진행합니다.

> **공개 범위는 어느 길로 가든 Public**
>
> 무료 계정에서 GitHub Pages로 주소를 만들려면 저장소가 공개여야 합니다. 비공개 저장소의 Pages는 유료 요금제 기능입니다. "공개"란 전 세계 누구나 내 코드를 읽을 수 있다는 뜻입니다. 그래서 127쪽의 안전 수칙이 중요합니다. 코드에 학생 이름·사진·API 키가 하나도 없어야 마음 놓고 공개할 수 있습니다.

5.9 AI Studio에서 코드 올리기

무대가 GitHub에서 AI Studio로 옮겨 갑니다. 4단원에서 만든 앱을 열고, 오른쪽 위 톱니바퀴를 눌러 패널을 연 뒤 GitHub 탭을 고릅니다. 패널 맨 위에는 탭이

한 줄로 놓여 있고, 화면이 좁으면 오른쪽 화살표로 넘겨야 나머지가 보입니다. 아직 연결하지 않았다면 "Sync to GitHub" 안내와 함께 새 저장소를 만드는 버튼이 나타납니다.

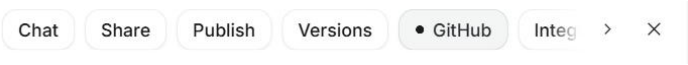


그림 5-7. 톱니바퀴를 누르면 열리는 패널의 탭 줄. 이 단원에서는 Share · Publish · GitHub 세 탭만 쓴다

1. 4단원에서 만든 앱을 엽니다. aistudio.google.com/apps → My apps에서 고릅니다.
2. 오른쪽 위 톱니바퀴를 클릭합니다. 패널이 열리고 Chat / Share / Publish / Versions / GitHub / Integrations / Secrets 탭이 보입니다.
3. GitHub 탭을 선택합니다. 탭이 많아 화살표로 넘겨야 보일 수 있습니다.
4. Create new repository를 누릅니다. 이미 만든 저장소가 있다면 그것을 고르는 항목도 있습니다.
5. GitHub 로그인과 권한 승인을 합니다. 처음 한 번만 하는 절차입니다.

권한 승인 화면이 낯설게 느껴질 수 있습니다. "AI Studio가 내 GitHub 저장소를 읽고 쓸 수 있게 한다"는 뜻이고, GitHub 설정에서 언제든 취소할 수 있습니다.

저장소 정보 입력—Public을 꼭 확인하세요

앞 절에서 본 github.com/new의 항목들이 여기에 요약되어 나타납니다. 이름·설명·공개 범위 세 가지만 정하면 됩니다.

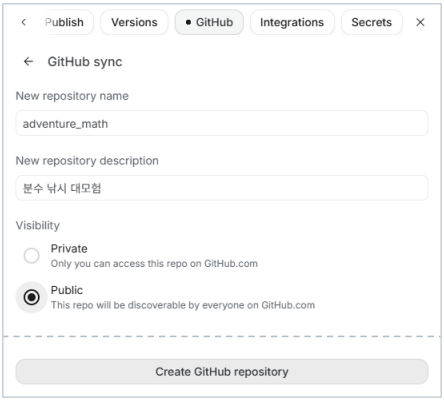


그림 5-8. GitHub 탭의 새 저장소 만들기. 이름·설명·Public 선택 후 Create

New repository name은 영문 소문자로 짓습니다. 이번 실습은 adventure_math이고, 이 이름이 주소 뒤에 그대로 붙습니다. New repository description은 한글로 적어 둡니다. Visibility에서 Public을 고릅니다. "This repo will be discoverable by everyone on GitHub.com"이라는 설명이 붙은 쪽입니다. Private을 고르면 무료 계정에서는 나중에 Pages를 켤 수 없어 되돌아와야 하니, 이 칸을 반드시 확인하세요.

Create GitHub repository를 누르면 "We're creating your repository" 안내가 뜨고, 잠시 뒤 동기화 화면으로 바뀝니다.

Push 하기—무엇이 올라가는지 보고 누르기

저장소가 만들어지면 패널이 동기화 화면으로 바뀝니다. 여기서 세 가지를 확인한 뒤 Push changes to GitHub를 누릅니다.

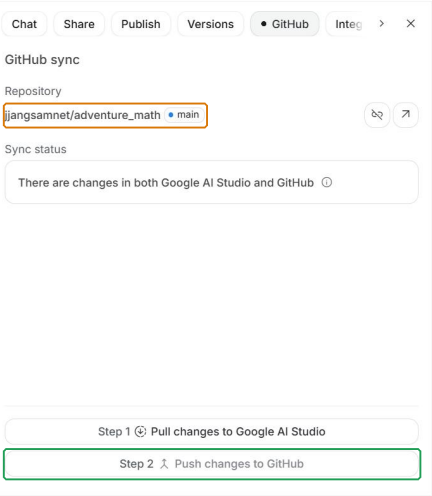


그림 5-9. GitHub sync 패널. 저장소(주황 표시) · 동기화 상태 · Push 버튼(초록 표시)

1. **저장소와 브랜치가 맞나요?** jjangsamnet/adventure_math처럼 "내 아이디 / 저장소 이름"이 표시되고 옆에 main 배지가 붙습니다. 이 두 값이 곧 나중에 만들 주소의 재료입니다.
2. **올릴 준비가 되었나요?** Sync status를 봅니다. "Changes in Google AI Studio are ready to be pushed"이면 올릴 것이 있다는 뜻이고, "There are changes in both …"이면 GitHub 쪽에도 변경이 쌓였다는 뜻이라 Step 1 Pull을 먼저 누릅니다. 아무 문구도 없으면 이미 올라간 상태입니다.
3. **커밋 메시지는 그대로.** AI가 "feat: initialize Fraction Fishing game project" 같은 문장을 자동으로 씁니다. 직접 고쳐도 되지만 그대로 두어도 무방합니다.

5.10 올라갔는지 눈으로 확인하기

Push를 눌렀다고 끝이 아닙니다. 브라우저에서 github.com의 내 저장소를 직접 열어 봅니다. 주소는 github.com/내아이디/저장소이름 입니다. 파일 목록에 index.html이 있으면 성공입니다. 이 파일이 최상위에 없으면 나중에 주소를 열었을 때 404가 뜹니다. 지금 확인하는 것이 가장 쌉니다.

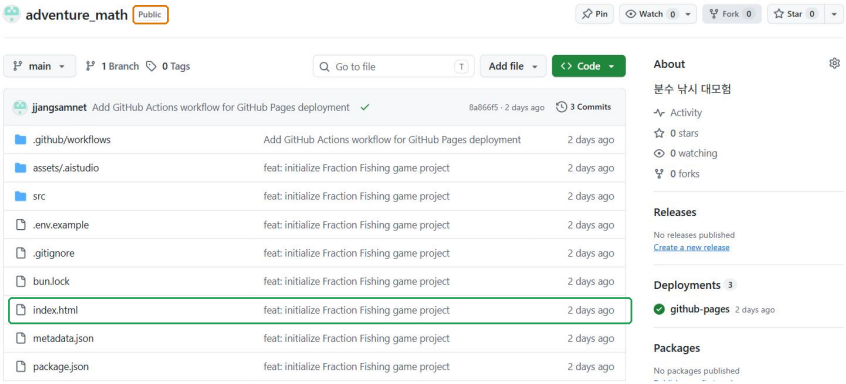


그림 5-10. 저장소 화면. 파일 목록에 index.html이 있고(초록 표시) 이름 옆에 Public 배지(주황 표시)가 보인다

1. 저장소 이름 옆 배지가 Public인지 봅니다. Private이면 무료 계정에서 Pages를 켤 수 없습니다.
2. 파일 목록에 index.html이 있는지 봅니다. Pages는 이 파일을 첫 화면으로 씁니다.
3. main 배지를 확인합니다. 기본 가지가 main 하나이고, 이 이름을 뒤에서 Pages 설정에 넣게 됩니다.
4. Commits 수를 봅니다. 저장 지점이 몇 개 쌓였다는 뜻이고, 누르면 기록을 볼 수 있습니다.

오른쪽 About 영역의 Deployments에 github-pages가 나타나면 이미 Pages가 켜져 배포까지 끝난 상태입니다. 아직 없다면 다음 절에서 켭니다. Add file과 Code 버튼은 파일을 직접 올리거나 코드를 내려받는 버튼인데, AI Studio가 대신 올려 주므로 쓸 일이 없습니다.

index.html을 열어보면—무엇이 들어 있나

파일 이름을 클릭하면 내용이 보입니다. 코드를 다 이해할 필요는 없습니다. 다만 "게임이 어떤 조각들로 되어 있는지" 한 번 보고 나면, 문제가 생겼을 때 AI에게 무엇을 고쳐 달라고 말해야 할지 감이 잡힙니다.

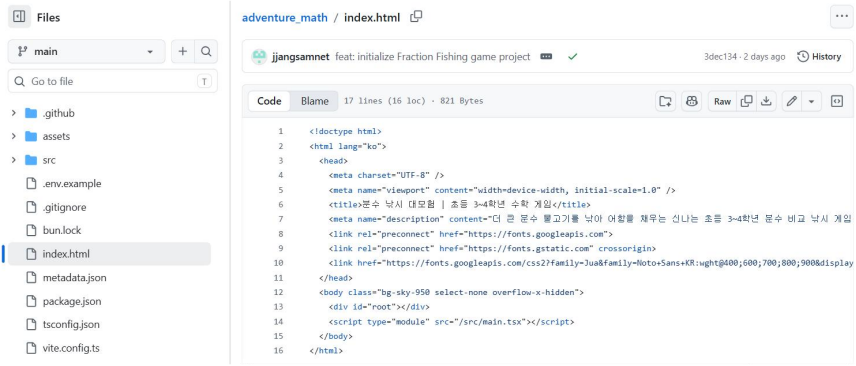


그림 5-11. index.html. 왼쪽에 파일 트리, 오른쪽에 코드

- <title>분수 낚시 대모험…</title> —브라우저 탭에 뜨는 제목. 여기가 한글이면 학생 화면 탭에도 한글로 보입니다.
- <meta name="description"> —검색·공유 시 보이는 한 줄 설명. AI가 게임 설명을 자동으로 넣어 두었습니다.
- —게임이 그려지는 빈 자리. 실제 화면은 아래 스크립트가 여기에 채웁니다.
- <script src="/src/main.tsx"> —게임 본체를 불러오는 줄. 게임 코드는 src 폴더 안에 있습니다.

커밋 기록—되돌릴 수 있다는 안심

저장소 화면의 “N Commits”를 누르면 목록이 나옵니다. 게임을 고칠 때마다 한 줄씩 쌓입니다. “AI에게 고쳐 달라고 했다가 더 나빠졌다”는 상황에서 이 기록이 안전망이 됩니다.

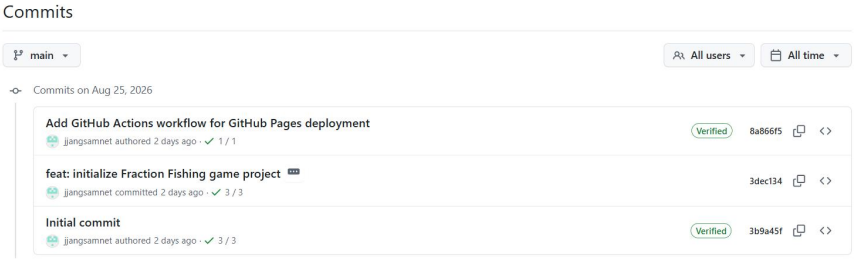


그림 5-12. 커밋 기록. 메시지·작성자·시각·Verified 배지·커밋 해시

커밋 메시지는 무엇을 바꿨는지 적은 한 줄이고 AI가 자동으로 씁니다. “Initial commit”이 첫 저장이며 그 위로 시간순으로 쌓입니다. Verified 배지는 서명이 확인된 커밋이라는 표시로, 교사 실습에서는 신경 쓰지 않아도 됩니다. 오른쪽의 여덟 자리 해시는 그 시점을 가리키는 고유 번호이고, 누르면 그때의 코드 전체를 볼 수 있습니다.

커밋 하나가 곧 되돌아갈 수 있는 지점 하나입니다. 크게 고치기 전에 한 번 올려 두면 마음이 편합니다.

5.11 Pages 켜기—주소가 태어나는 곳

코드가 올라갔다고 해서 자동으로 주소가 생기지는 않습니다. Pages 기능을 한 번 켜 줘야 합니다. 저장소 화면 맨 위 탭 줄의 오른쪽 끝 Settings를 누르고, 왼쪽 사이드바를 아래로 내리면 Pages가 있습니다.

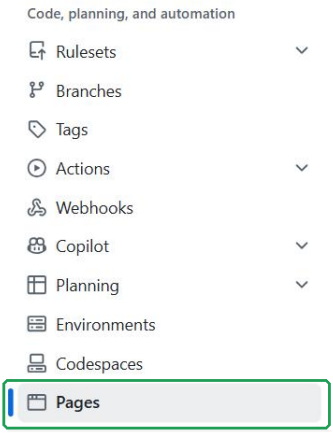


그림 5-13. Settings 왼쪽 사이드바. “Code, planning, and automation” 묶음 아래에 Pages(초록 표시)

사이드바는 General → Access → Code, planning, and automation → Security and quality → Integrations 순서로 묶여 있습니다. **Pages**는 세 번째 묶음의 아래쪽, Environments·Codespaces 다음에 있습니다. 찾기 어렵다면 주소창에 저장소 주소 뒤 **/settings/pages**를 직접 붙여도 같은 곳으로 갑니다. 참고로 이 화면 맨 아래에는 **Danger Zone**이 있습니다. 126쪽에서 회수 절차로 다시 옵니다.

Source —게시 방식 두 갈래

Pages 화면의 Build and deployment → Source에서 게시 방식을 고릅니다. 드롭다운을 열면 두 가지가 나옵니다. 어느 쪽을 골라도 최종 주소는 같습니다. 다만 실습 중에 보게 되는 화면과 확인 방법이 달라지니, 지금 차이를 알고 가세요.

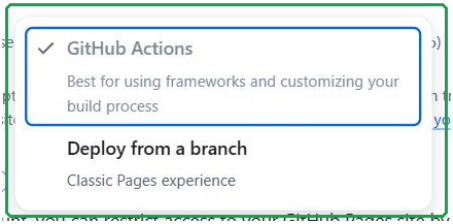


그림 5-14. Source 드롭다운을 열면 나오는 두 가지. 위가 GitHub Actions, 아래가 Deploy from a branch

Deploy from a branch는 "Classic Pages experience"입니다. 가지(branch)와 폴더를 지정하면 그 안의 파일을 그대로 올려 줍니다. 완성된 index.html이 이미 있는 경우에 가장 단순합니다. GitHub Actions는 "Best for using frameworks and customizing your build process"로, 저장소에 들어 있는 배포 절차서(workflow)를 실행해 만들어진 결과물을 올립니다. AI Studio가 이 절차서를 함께 올려 두면 자동으로 이쪽이 선택되어 있습니다.

> **어느 쪽이 이미 선택되어 있나요?**
>
> AI Studio로 Push했다면 GitHub Actions가 켜져 있을 가능성이 큽니다. 저장소에 .github/workflows 폴더가 보였다면 배포 절차서가 함께 올라간 것입니다. 이미 잘 배포되고 있다면 굳이 바꾸지 마세요. 이어지는 "첫째 길"은 Branch가 None이고 아무것도 배포되지 않은 경우를 위한 안내입니다.

첫째 길 · Deploy from a branch

처음 들어가면 Branch가 None으로 되어 있고 “GitHub Pages is currently disabled”라고 적혀 있습니다.

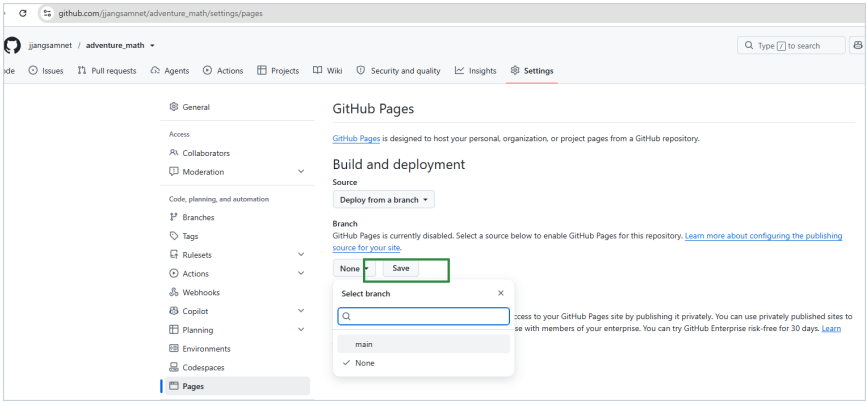


그림 5-15. Settings → Pages. Branch 드롭다운에서 main 선택(초록 표시)

1. Source에서 Deploy from a branch를 고릅니다.
2. Branch의 None을 눌러 목록을 열고 main을 선택합니다. 저장소가 비어 있으면 main이 보이지 않습니다. 먼저 Push하세요.
3. 폴더는 / (root)로 둡니다. index.html이 최상위에 있으므로 그대로입니다. 하위 폴더에 있다면 그 폴더를 고릅니다.
4. Save를 누릅니다. 저장하면 배포가 시작됩니다.
5. 1–2분 뒤 새로고침하면 화면 위쪽에 초록 안내와 주소가 나타납니다.

> **발급되는 주소의 형식**
>
> https://<내 GitHub 사용자명>.github.io/<저장소 이름>/
>
> 예) https://jjangsamnet.github.io/adventure_math/

둘째 길 · GitHub Actions

Source가 GitHub Actions로 되어 있으면 Push할 때마다 배포 절차서가 자동으로 돕니다. 따로 Save를 누를 것이 없고, 대신 저장소 상단의 Actions 탭에서 배포가 잘 돌았는지 확인합니다.

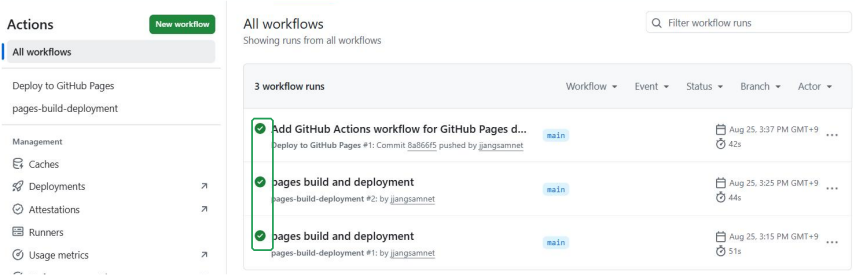


그림 5-16. Actions 탭. 워크플로 실행 목록에서 초록 체크가 배포 성공

왼쪽 목록의 Deploy to GitHub Pages, pages build and deployment가 배포 절차서의 이름입니다. 초록 체크는 그 실행이 성공했다는 뜻이고, 옆에 시각과 소요 시간이 함께 보입니다. 노란 원이면 아직 배포가 돌고 있는 중이니 보통 1–2분 기다렸다 새로고침합니다. 빨간 X가 뜬다면 실행을 눌러 들어가 어느 단계에서 멈췄는지 로그를 확인하고, 그 문장을 그대로 AI에게 붙여넣고 물어보세요.

"Your site is live at …" —주소가 태어난 순간

Pages 화면을 새로고침하면 위쪽에 초록 안내와 함께 주소가 나타납니다.

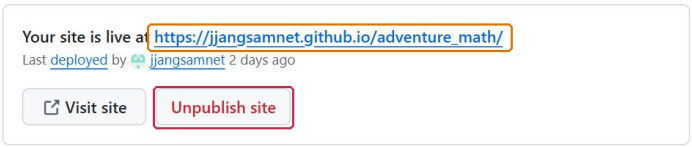


그림 5-17. Pages 배포 완료. 발급된 주소(주황 표시)와 Visit site · Unpublish site(분홍 표시)

Your site is live at …가 발급된 주소이고, 학생에게 나눠 줄 한 줄입니다. Visit site는 새 탭에서 바로 열어 봅니다. 오른쪽의 Unpublish site는 게시를 끄는 버튼으로, 문제가 생겼을 때 가장 빠른 회수 방법입니다. Last deployed by …는 마지막 배포 시각이며, 고쳤는데 반영이 안 될 때 여기부터 봅니다. Custom domain

은 학교 도메인이 있다면 붙일 수 있고, 없으면 비워 둡니다. Enforce HTTPS는 기본 도메인에서 자동으로 켜지고 끌 수 없습니다. 안전한 상태라는 뜻입니다.

접속해서 확인하기

주소를 새 탭에 붙여넣어 게임이 실제로 열리는지 확인합니다. 이 순간이 이 단원의 도착점입니다. 이제 이 한 줄만 있으면 누구나, 어떤 기기에서든 내 게임을 열 수 있습니다. 스마트폰으로도 꼭 한 번 열어 보세요.

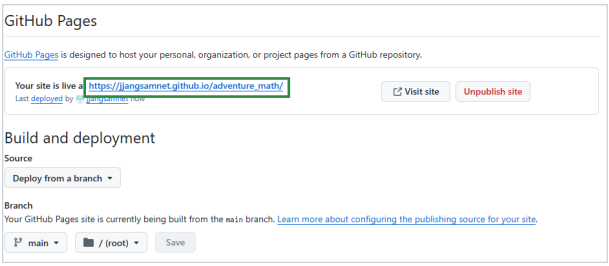


그림 5-18. 발급된 주소로 접속한 게임. 주소창을 함께 확인한다

> **안 될 때 세 가지**
>
> **404가 뜬다**—index.html이 저장소 최상위에 있는지 확인합니다(116쪽). 하위 폴더라면 파일을 최상위로 옮기고 다시 Push합니다.
>
> **옛날 화면이 뜬다**—브라우저 캐시입니다. Ctrl+F5 또는 시크릿 창으로 확인하세요. 배포 반영에 1–2분이 걸립니다.
>
> **화면이 깨진다**—이미지·소리 경로가 내 PC 기준일 수 있습니다. AI Studio에 "모든 자원을 상대 경로로 바꿔줘"라고 요청하세요.

5.12 학생에게 닿게 하기

주소가 있어도 학생 손에 닿지 않으면 소용없습니다. 주소는 길고, 초등학생이 타이핑하기에는 오타가 납니다. 실제 교실에서는 "타이핑하지 않고 도달하는 경로"를 반드시 마련해야 합니다. 아래 네 가지 중 학교 환경에 맞는 것을 두 개 이상 겹쳐 두면 수업 중 사고가 거의 사라집니다.

- **QR 코드** —주소를 QR로 만들어 TV 화면에 띄우거나 학습지 귀퉁이에 인쇄합니다. 태블릿 카메라로 바로 접속합니다. 가장 빠르고 인쇄가 됩니다.
- **학급 플랫폼 링크** —클래스룸·e학습터·학급 밴드 게시글에 링크를 겁니다. 가정 연계 과제로 낼 때 필수 경로입니다.
- **북마크 사전 설치** —교실 공용 태블릿이라면 수업 전에 브라우저 즐겨찾기에 미리 등록해 둡니다. 저학년일수록 효과가 큽니다.
- **짧은 주소** —주소가 길면 단축 서비스로 줄여 칠판에 적습니다. 다만 학교 망에서 단축 도메인이 막히는 경우가 있어 사전 확인이 필요합니다.

겹쳐 두라는 것은 예를 들어 QR(교실용)과 클래스룸 링크(가정용)를 함께 준비하라는 뜻입니다. 태블릿 카메라가 안 되는 기기가 한둘 나오고, 집에서는 QR을 찍을 화면이 없습니다.

QR 코드 만들기—5분이면 됩니다

QR 코드는 별도 프로그램 없이 만들 수 있습니다. 만드는 것보다 중요한 것은 "어디에 얼마나 크게 넣느냐"입니다.

1. QR 생성 사이트를 엽니다. "QR 코드 생성"으로 검색하면 무료 사이트가 많이 나옵니다. 로그인이 필요 없고 광고가 적은 곳을 고르세요.
2. 주소를 붙여넣습니다. https://부터 마지막 /까지 전부 복사해 넣습니다. 한 글자라도 빠지면 다른 곳으로 갑니다.
3. PNG로 내려받습니다. 가능하면 크게(1000픽셀 이상) 받아 두세요. 학습지에 인쇄할 때 작게 받으면 뭉개집니다.
4. 교실 TV용은 화면 가득. 수업 시작 시 TV에 QR만 크게 띄웁니다. 학생들이 자리에서 찍을 수 있는 크기여야 합니다.
5. 학습지용은 한 변 3 cm 이상. 그보다 작으면 인식률이 떨어집니다.
6. 만든 QR을 내 스마트폰으로 한 번 찍어 봅니다. 이 한 번이 수업 중 사고를 막습니다.

> **주소를 짧게 만들고 싶다면**
>
> 저장소 이름을 짧게 지으면 주소도 짧아집니다. adventure_math 대신 math4처럼요. 이름은 나중에 Settings → Rename으로 바꿀 수 있지만, 바꾸면 예전 주소가 끊깁니다. 이미 학생에게 나눠 준 뒤라면 바꾸지 마세요. 단축 서비스는 학교 망에서 막히는 사례가 있으니 보조 수단으로만 쓰세요.

수업 전날 체크리스트
교실 기기 한 대에서 주소가 열리는지 확인 → QR 이미지를 만들어 학습지에 넣기 → 클래스룸 게시글 예약 발행 → 예비용으로 짧은 주소도 준비.

5.13 수업 배치—이 게임을 언제 꺼내는가

주소가 생겼다고 게임이 저절로 수업이 되지는 않습니다. 언제 꺼내느냐에 따라 같은 게임이 진단 도구가 되기도 하고 단순한 시간 때우기가 되기도 합니다. 아래 세 가지 배치를 각각 한 번씩 해 보고 우리 반에 맞는 것을 고르세요.

도입 5분·진단

단원 시작에 붙여 사전 지식을 확인합니다. 점수는 기록하지 않고 "오늘 뭘 배울지 맛보기"라고 말해 둡니다. 교사는 순회하며 어디서 많이 막히는지만 봅니다. 5분을 넘기지 않습니다. 길어지면 본 수업이 밀립니다.

정리 10분·형성평가

오늘 배운 개념을 게임으로 되짚습니다. 가장 효과가 큰 배치입니다. 방금 배운 것을 바로 써 보게 하고, 교사는 순회하며 개별 지도할 학생을 파악합니다. 다 못 푼 학생에게는 "집에서 더 해도 된다"고 링크를 안내합니다.

가정·자율 반복

링크를 클래스룸에 걸어 원하는 만큼 반복하게 합니다. 강제 과제로 만들지 않는 편이 참여율이 높습니다. "해 온 사람 손 들어"를 하지 않습니다. 안 한 학생이 드러납니다. 기기가 없는 가정을 반드시 고려하고 대체 활동을 함께 준비하세요.

공통 주의
어느 배치든 "게임 점수를 성적·상점·자리 배치에 연결하지 않기"가 원칙입니다. 연

> 결하는 순간 학생은 학습이 아니라 점수를 최적화합니다. 반복하면 오르는 구조라는 점도 미리 말해 두세요. 6단원 리더보드에서 이 이야기를 본격적으로 다룹니다.

5.14 고치고 다시 올리기, 그리고 회수하기

수업을 해 보면 반드시 고칠 곳이 나옵니다. "문제가 너무 어려워요", "글씨가 작아요". 한 번 연결해 두었으면 그다음부터는 간단합니다. AI Studio에서 고치고, Push 한 번. 학생이 보는 주소는 그대로입니다.

1. AI Studio에서 고칩니다. 4단원에서 하던 대로 채팅으로 요청합니다. 예: "3단계 문제를 조금 쉽게 바꿔줘"
2. GitHub 탭에서 Push합니다. 변경 파일 수가 다시 표시되면 Push changes to GitHub를 누릅니다.
3. Actions에서 배포를 확인합니다. 초록 체크가 뜰 때까지 1–2분. Deploy from a branch를 쓰고 있다면 자동으로 다시 배포됩니다.
4. 주소를 강력 새로고침합니다. Ctrl+F5. 학생 기기에도 캐시가 남아 있을 수 있으니 수업 전에 미리 갱신해 두세요.

주소는 바뀌지 않습니다. 이것이 GitHub Pages의 가장 큰 장점입니다. 몇 번을 고쳐 올려도 학생에게 나눠 준 주소는 그대로이고, 이미 인쇄한 QR 학습지도 계속 유효합니다. 잘못 고쳤다면 커밋 기록(116쪽)에서 이전 시점을 찾을 수 있고, AI Studio 패널의 Versions 탭에서도 이전 버전으로 돌아갈 수 있습니다.

> 수업 중에는 고치지 마세요. 배포에 1–2분이 걸리고 그동안 화면이 흔들립니다.

회수하기—문제가 생겼을 때 즉시 닫는 법

공개한 자료에 문제가 생기는 일은 실제로 일어납니다. 실수로 학생 이름이 들어갔거나, 저작권 있는 이미지를 썼거나. 중요한 것은 속도입니다. 두 가지 방법을 미리 알아 두세요. 둘 다 1분이면 됩니다.

첫째, Unpublish site. 가장 빠르고 권장하는 방법입니다. Settings → Pages

화면 오른쪽 위의 Unpublish site를 누릅니다. 주소가 즉시 닫히고 접속하면 404가 뜹니다. 저장소와 코드는 그대로 남아 있으므로, 고친 뒤 다시 켜면 같은 주소가 살아납니다.

둘째, 저장소를 Private으로 전환. Settings → General 맨 아래 Danger Zone → Change visibility. 주소가 닫히는 동시에 코드도 남에게 보이지 않게 됩니다. 코드 자체에 문제가 있을 때(개인정보·API 키) 이쪽을 씁니다.

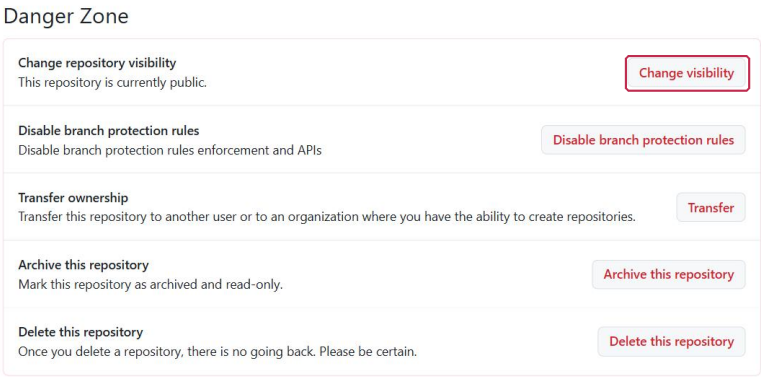


그림 5-19. Settings → General 맨 아래 Danger Zone. 분홍 표시가 공개 범위 전환

> Danger Zone에는 Transfer(소유권 이전)·Archive(읽기 전용)·Delete(영구 삭제)도 있습니다. Delete는 되돌릴 수 없으니 절대 급하게 누르지 마세요.

5.15 공개하기 전 3분 점검

GitHub Pages로 주소를 만들었다는 것은 그 코드가 인터넷에 공개되었다는 뜻입니다. 게시 버튼을 누르기 전, 아래 여섯 칸을 소리 내어 읽으며 점검하는 습관을 들이세요. 한 번의 실수가 되돌리기 어려운 결과로 이어질 수 있습니다.

- ☐ **학생 이름·학번** —문항이나 예시에 실제 학생 이름, 학번, 반 번호가 들어가 있지 않은지. 이름이 필요하면 "친구 1, 친구 2"로 바꿉니다.

- ☐ **사진·음성 파일** —학생 얼굴 사진이나 목소리가 게임 자원에 포함되어 있지 않은지. 초상권과 개인정보 문제가 동시에 걸립니다.
- ☐ **API 키·비밀번호** —코드 안에 API 키가 문자열로 박혀 있지 않은지. 공개 저장소의 키는 자동 수집 대상이 됩니다.
- ☐ **저작권 자원** —인터넷에서 가져온 이미지·음악·캐릭터가 들어 있지 않은지. 수업용이라도 공개 게시는 사적 이용 범위를 벗어납니다.
- ☐ **저장소 설명** —설명에 학교명·학급명을 그대로 적지 않았는지. "3학년 수학 게임" 정도면 충분합니다.
- ☐ **회수 절차 숙지** —126쪽의 Unpublish site와 Private 전환을 미리 한 번 눌러 보세요. 급할 때 찾느라 헤매지 않습니다.

법적 기준

만 14세 미만 아동의 개인정보를 처리하려면 법정대리인의 동의가 필요합니다(개인정보 보호법 제22조의2).[a] 초등학교 수업에서 만든 게임은 애초에 개인정보를 수집하지 않는 설계가 원칙입니다. 점수·기록이 필요하다면 학생이 직접 정한 닉네임만 쓰게 하세요. 6단원에서 그 방법을 다룹니다.

[a]개인정보보호위원회. (2023). *개인정보 보호법 제22조의2(만 14세 미만 아동의 개인정보 보호).*; 교육부. (2024). *교육(지원)청·학교 개인정보 보호 가이드라인.*

5.16 100분 미션—내 게임에 주소 달기

4단원에서 만든 게임을 그대로 재료로 씁니다. 그림을 옆에 두고 순서대로 따라오세요. 마지막 산출물은 "동료에게 보낼 수 있는 링크 한 줄"과 "학습지에 붙일 QR 한 장"입니다. 연수에서는 이 100분을 두 차시로 나누어 운영하기를 권합니다.

0–10분 **내 게임 진단**
AI 기능이 있는지 확인하고 Publish 길인지 Pages 길인지 결정합니다(103쪽).

10–18분 **공유 링크 만들기**
Share로 옆자리 선생님에게 링크를 전달하고, 상대 화면에서 열리는지 확인합니다(104쪽).

18 – 28분 **Cloud Run 게시**
Publish → Get started → Publish App으로 발급된 주소에 접속합니다(106쪽).

28 – 43분 **GitHub 익히기**
계정을 준비하고 첫 화면·저장소 목록·새 저장소 폼을 눈으로 확인합니다(109쪽).

43 – 58분 **코드 올리기**
AI Studio GitHub 탭에서 Public 저장소를 만들고 Push한 뒤, github.com에서 index.html을 확인합니다(112쪽).

58 – 76분 **Pages 켜기**
Settings → Pages에서 Source를 확인하고 주소를 발급받아 접속에 성공합니다(119쪽).

76 – 90분 **학생 배포 준비**
QR을 만들어 학습지 하단에 넣고 스마트폰으로 실제 접속을 테스트합니다(123쪽).

90 – 100분 **회수 연습과 안전 점검**
Unpublish site를 눌러 닫아 보고 다시 켠 뒤, 여섯 칸 점검표를 소리 내어 읽습니다(127쪽).

제출 양식

내 게임 이름: _____

선택한 경로: ☐ Publish(Cloud Run) ☐ GitHub Pages

저장소 이름: _____

발급된 주소: _____

접속 확인: ☐ PC ☐ 스마트폰 ☐ 교실 태블릿

QR 제작: ☐ 회수 연습: ☐

5.17 자주 막히는 지점 여덟

연수 현장에서 실제로 가장 많이 나온 질문들입니다. 대부분은 "설정을 하나 빠뜨렸다"거나 "기다림이 부족했다"에 해당합니다.

Q. Branch에 main이 안 보여요

A. 아직 Push가 되지 않아 저장소가 비어 있는 경우입니다. AI Studio에서 Push changes to GitHub를 먼저 누르고, Pages 화면을 새로고침하세요(112쪽).

Q. 주소를 열었더니 404가 떠요

A. 저장소 최상위에 index.html이 있어야 합니다(116쪽). 하위 폴더에 있다면 AI Studio에 "index.html을 최상위로 옮겨줘"라고 요청한 뒤 다시 Push하세요.

Q. Pages 메뉴가 안 보여요

A. 저장소가 Private일 때 무료 계정에서는 제한됩니다. Settings → Danger Zone에서 Public으로 바꾸면 나타납니다(126쪽).

Q. 게임은 뜨는데 AI 기능이 안 돼요

A. 정상입니다. GitHub Pages는 정적 호스팅이라 서버가 없어 AI 호출이 작동하지 않습니다. AI 기능이 필요하면 Publish(Cloud Run) 경로를 쓰세요(103쪽).

Q. 고쳤는데 주소에 반영이 안 돼요

A. Push를 눌렀는지, Actions에 초록 체크가 떴는지, Ctrl+F5로 강력 새로고침했는지 순서대로 확인합니다. 배포에 1–2분이 걸립니다(126쪽).

Q. Source가 이미 GitHub Actions예요

A. AI Studio가 배포 절차서를 함께 올린 것입니다. 잘 배포되고 있다면 바꾸지 마세요. Actions 탭에서 결과만 확인하면 됩니다(119쪽).

Q. Publish에서 결제하라고 나와요

A. 이미 Google Cloud 결제 계정을 쓰고 있는 계정은 Starter Tier 대상이 아닙니다. Cloud 프로젝트를 연결하거나, AI 기능이 없다면 Pages로 우회하세요.

Q. 학교 망에서 안 열려요

A. 교육청 망 필터링일 가능성이 큽니다. 수업 전날 반드시 교실 기기 한 대로 접속 테스트를 하고, 막히면 정보부장이나 교육청에 도메인 허용을 요청하세요.

5.18 단원 마무리

이제 선생님의 게임은 선생님 컴퓨터를 떠났습니다. 링크 한 줄이면 교실 태블릿에서도, 학생의 집 스마트폰에서도 열립니다.

이 단원에서 우리는 출판 방법 네 가지를 비교해 지도를 폈고, "실행 중에 AI에게 말을 거는가"라는 질문 하나로 내 게임의 길을 갈랐습니다. 그다음 GitHub의 계정·저장소·커밋·공개 범위를 화면을 눌러 가며 익히고, AI Studio에서 코드를 Push한 뒤 github.com에서 파일이 실제로 올라갔는지 눈으로 확인했습니다. Settings → Pages에서 게시 방식을 고르고 주소를 발급받았으며, QR과 학급 플랫폼으로 학생에게 닿게 하는 경로를 마련했습니다. 마지막으로 고쳐 올리는 절차와 즉시 닫는 절차까지 손에 익혔습니다.

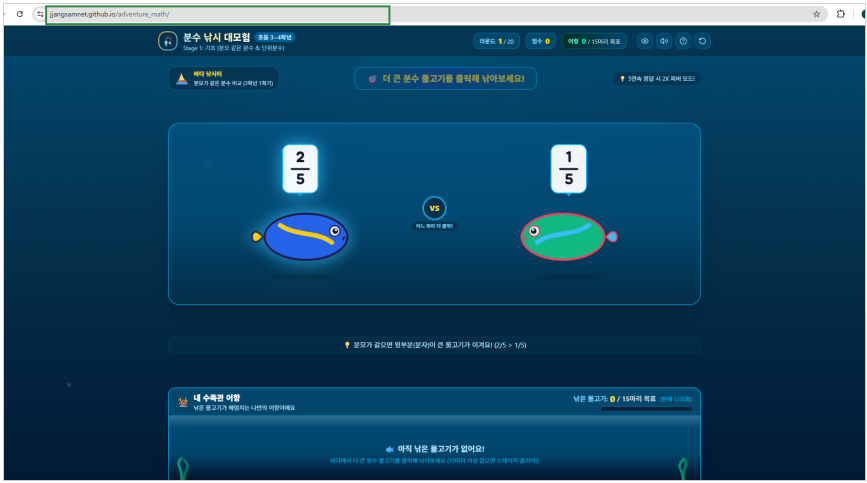


그림 5-20. 발급된 주소로 열린「분수 낚시 대모험」. 이 단원의 도착점

남은 숙제. 만든 주소를 다음 수업 학습지 하단에 QR로 넣어 보고, 실제 학생 반응을 한 문장으로 기록해 두세요. 6단원 리더보드 설계의 재료가 됩니다.

다음 단원 예고. WORLD 6 ·실시간 리더보드—Firebase Firestore를 붙여 TOP 10과 내 순위를 보여주고, 교사용 회차 초기화까지 만듭니다.

주소를 가진 게임은 더 이상 내 자료가 아니라 우리 반의 자료입니다.

참고문헌

Google. (2026). *Build apps in Google AI Studio.* Gemini API 개발자 문서. https://ai.google.dev/gemini-api/docs/aistudio-build-mode

Google. (2026). *Deploying from Google AI Studio.* Gemini API 개발자 문서. https://ai.google.dev/gemini-api/docs/aistudio-deploying

GitHub. (2026). *What is GitHub Pages? / GitHub Pages limits.* GitHub Docs. https://docs.github.com/pages

GitHub. (2026). *Configuring a publishing source for your GitHub Pages site.* GitHub Docs. https://docs.github.com/pages

교육부. (2024). *교육(지원)청·학교 개인정보 보호 가이드라인.*

개인정보보호위원회. (2023). *개인정보 보호법 제22조의2(만 14세 미만 아동).*

한국콘텐츠진흥원. (2025). *2025 게임이용자 실태조사.*

화면 캡처: GitHub(github.com) 및 Google AI Studio, 2026년 8월. 서비스 정책에 따라 화면 구성은 변경될 수 있습니다.

WORLD 6

실시간 리더보드

우리 반의 기록이 서로에게 보이도록

5단원까지 선생님의 게임에는 주소가 생겼습니다. 그런데 학생 서른 명이 각자 자기 화면에서 점수를 얻어도, 그 기록은 서로에게 보이지 않습니다. 이 단원은 그 벽을 허뭅니다.

Firebase Firestore를 붙여 우리 반의 기록을 한곳에 모으고, 실명 없이 회차별로 보이는 순위표를 만듭니다. 그리고 그 데이터베이스 앞에 문지기—보안 규칙—를 세우는 일까지가 이 단원의 몫입니다.

이 단원은 앞의 다섯 단원보다 두 배 깁니다. Firebase 콘솔의 실제 화면을 한 장씩 따라가며 "어디를 눌러야 하는지"를 빠짐없이 담았기 때문입니다. 본문의 그림은 2026년 8월에 실제 Firebase 콘솔(한국어 UI)과 Google AI Studio에서 촬영한 것이며, 개인정보에 해당하는 이메일과 키 값은 흐리게 처리했습니다. 구글 정책에 따라 버튼의 위치나 이름이 바뀔 수 있으니, 그림과 다르면 같은 뜻의 메뉴를 찾으시면 됩니다.

이 단원의 도착점은 다음 한 줄입니다.

> **이 단원의 도착점**
>
> 실명 없이, 회차별로, 내 기록과 나란히 보이는 우리 반 순위표.
>
> 옆자리 선생님의 기기에서 점수를 올리면 내 화면의 순위표가 새로고침 없이 바뀌는 장면—그것이 실습을 마친 증거입니다.

6단원 퀘스트—무엇을 얻어 가나요

리더보드는 기술적으로는 30분이면 붙습니다. 어려운 것은 그다음입니다. 누구의 이름을 띄울 것인가, 몇 등까지 보여줄 것인가, 언제 지울 것인가. 그리고 "내가 만든 데이터베이스는 지금 안전한가". 이 단원의 절반은 그 판단에 씁니다.

- **순위표를 교육적으로 설계한다** —순위표가 동기를 낮추기도 한다는 연구를 바탕으로, 교실에서 지킬 설계 원칙 다섯 가지를 먼저 정합니다.
- **Firebase 콘솔을 혼자 연다** —프로젝트 만들기부터 Firestore, 인증, 보안 규칙까지 실제 화면을 눌러 가며 익힙니다.
- **데이터를 눈으로 확인한다** —컬렉션·문서·필드가 콘솔에서 어떻게 보이는지, 개인정보가 섞여 있지 않은지 직접 점검합니다.
- **로그인을 걷어낸다** —기본으로 켜지는 Google 로그인을 익명 로그인과 닉네임으로 바꿔, 개인정보를 아예 수집하지 않는 구조로 만듭니다.
- **보안 규칙을 읽고 고친다** —"allow read, write: if true"가 무슨 뜻인지 알아보고, 점수 범위와 수정 금지를 직접 넣습니다.
- **교사용 회차 초기화를 만든다** —기록을 지우지 않고 새 회차를 여는 방식으로, 매 수업 깨끗한 순위표에서 시작합니다.

8단계 리더보드 로드맵—총 100분

5단원까지 만든 게임을 그대로 재료로 씁니다. 앞의 두 단계(설계 판단과 개념 이해)를 건너뛰고 바로 만들면, 만들고 나서 지우게 됩니다. 실제 연수에서 가장 많이 나온 후회가 "실명으로 만들었다가 다시 뜯어고쳤다"입니다. 순서대로 가세요.

단계	하는 일	핵심
STEP 1	Firestore 이해 (8분)	클라우드 칠판 비유
STEP 2	프로젝트 만들기 (12분)	콘솔 3단계 마법사
STEP 3	Firestore 열기 (15분)	컬렉션·문서·필드
STEP 4	익명 인증 켜기 (12분)	로그인 걷어내기
STEP 5	보안 규칙 (18분)	내 규칙 검사 → 수정
STEP 6	앱에 붙이기 (10분)	Enable Firebase
STEP 7	리더보드 구현 (15분)	TOP 10 + 내 순위
STEP 8	회차 초기화 (10분)	교사용 패널

먼저 읽고 가세요

STEP 4(익명 인증)를 뒤로 미루면 안 됩니다. Firebase 인증은 기본값이 Google 로그인이고, 초등학생 계정으로 로그인시키는 순간 만 14세 미만 개인정보 문제가 그대로 걸립니다. 그리고 STEP 5(보안 규칙)는 절대 건너뛸 수 없는 단계입니다. 그 이유를 160쪽에서 실제 화면으로 보여드립니다.

6.1 순위표는 약인가 독인가—만들기 전에 정할 것

게이미피케이션 연구는 리더보드에 대해 한 방향으로 말하지 않습니다. Hanus와 Fox(2015)는 배지와 리더보드를 넣은 수업에서 한 학기 동안 내재적 동기와 만족도, 시험 점수가 오히려 낮아지는 것을 관찰했습니다.[1] 도구가 나쁜 것이 아니라, 순위표가 "남과의 비교"를 학습의 중심에 놓을 때 그렇게 됩니다.

그래서 이 단원은 만드는 법보다 정하는 일을 먼저 합니다. 아래 다섯 가지는 그 위험을 줄이는 설계이고, 나중에 AI에게 리더보드를 만들어 달라고 요청할 때 그대로 프롬프트의 문장이 됩니다(167쪽). 우리 반에 그대로 적용할 것과 바꿀 것을 지금 표시해 두세요.

하나, 실명 금지—닉네임만

개인정보 문제이자 낙인 문제입니다. "3반 김OO 28위"가 화면에 뜨는 순간, 그 학생에게 이 게임은 벌이 됩니다. 순위표에 필요한 것은 신원이 아니라 서로를 구분할 표식 하나뿐입니다.

둘, 회차마다 초기화

누적 순위는 만들지 않습니다. 순위표의 수명은 한 차시입니다. 어제 진 학생이 오늘 다시 0에서 시작할 수 있어야 합니다. 이 원칙을 기록 삭제가 아니라 새 회차 코드 발급으로 구현하는 방법은 169쪽에서 다룹니다.

셋, 내 기록을 나란히

"우리 반 1위 820점" 옆에 "내 최고 기록 340 → 410점"을 함께 띄웁니다. 비교 대상을 남에서 어제의 나로 옮기는 장치입니다. 다섯 가지 중 화면 설계에 가장 직접적으로 영향을 주는 항목이므로, 프롬프트에서도 빠뜨리지 마세요.

[1] Hanus, M. D., & Fox, J. (2015). Assessing the effects of gamification in the classroom. *Computers & Education, 80,* 152–161.

넷, TOP 10까지만 공개

전체 순위는 공개하지 않습니다. 상위권은 보이게 하되 하위권은 남에게 보이지 않게 합니다. 자기 순위는 자기 화면에만 뜨게 하는 것이 원칙입니다.

다섯, 점수는 성취가 아니다

반복하면 오르는 구조임을 학생에게 먼저 말해 둡니다. 순위를 평가나 상점, 자리 배치에 연결하는 순간 설계가 무너집니다. 이것은 5단원의 수업 배치 원칙과 같은 이야기이고, 데이터베이스가 붙어 기록이 남기 시작하면 더 중요해집니다.

> **이론적 배경**
>
> 숙달목표(내가 어제보다 나아졌는가)와 수행목표(내가 남보다 나은가)는 학습에 다른 결과를 냅니다(Ames, 1992).[a] 셋째 원칙은 숙달목표를 화면에 남겨 두기 위한 장치입니다. 순위표를 아예 쓰지 말자는 이야기가 아니라, 화면 안에 두 목표를 함께 두자는 이야기입니다.
>
> [a]Ames, C. (1992). Classrooms: Goals, structures, and student motivation. *Journal of Educational Psychology, 84*(3), 261–271.

6.2 길이 바뀝니다—이제 서버가 필요합니다

5단원에서 우리는 "이 게임은 실행 중에 AI에게 말을 거는가?"라는 질문 하나로 길을 갈랐습니다. 리더보드는 AI를 부르지는 않지만, 여러 사람의 기록을 한곳에 모아 두고 서로 읽어야 합니다. 즉 저장소가 필요합니다. 질문을 한 칸 넓혀 봅시다.

새 질문은 이렇습니다. "내 게임은 내 화면 밖의 무언가가 필요한가?" 여기서 화면 밖의 무언가란 AI 호출이거나 공용 데이터입니다. 둘 중 하나라도 해당하면 서버 쪽으로 넘어갑니다.

아니오—코드만으로 완결되는 게임

5단원의「분수 낚시 대모험」이 여기에 해당합니다. 문항과 규칙이 코드 안에 있고 점수는 내 화면에만 남습니다. 이런 게임은 **GitHub Pages**로 충분합니다. 무료이고 기한이 없으며 내 계정 소유입니다. 이미 만들어 둔 주소는 그대로 두면 됩니다.

예—공용 저장소가 필요한 게임

리더보드가 붙은 순간 여기로 넘어옵니다. 남의 점수를 읽고 내 점수를 남기려면 공용 저장소가 있어야 합니다. 이때의 길은 **Firestore + Publish(Cloud Run)**입니다. Starter Tier로 결제 없이 앱 두 개까지 게시할 수 있고, Firestore와 Authentication, Cloud Run이 한 세트로 붙습니다.

> **GitHub Pages 주소는 어떻게 되나**
>
> 5단원에서 만든 GitHub Pages 주소는 그대로 살아 있습니다. 다만 리더보드가 붙은 버전은 그 주소에서 순위표만 비어 보입니다. 정적 호스팅에는 데이터를 담을 곳이 없기 때문입니다. 리더보드 버전의 최종 주소는 Publish로 발급되는 Cloud Run 주소이고, 학생에게는 그쪽을 나눠 줍니다. 두 주소가 동시에 살아 있어도 문제가 되지 않으니, 어느 쪽을 나눠 줬는지만 헷갈리지 않게 적어 두세요.

6.3 Firestore —교실 뒤에 걸린 클라우드 칠판

Firestore는 구글이 운영하는 데이터베이스입니다. 교사에게 필요한 이미지는 하나입니다. 전국 어디서든 보이는 칠판이 하나 걸려 있고, 누가 거기에 무엇을 적으면 그 칠판을 보고 있던 모든 사람의 화면이 즉시 바뀝니다. "새로고침을 누르지 않아도 바뀐다"가 실시간(real-time)의 뜻입니다.

용어는 세 개면 충분합니다. 이 셋만 알면 콘솔 화면에서 길을 잃지 않습니다.

컬렉션(Collection) —같은 종류의 기록을 모아 두는 상자

칠판에 붙인 "점수" 코너라고 생각하면 됩니다. 우리는 scores라는 상자 하나만 씁니다. 컬렉션은 여러 쪽지가 들어가는 상자이므로 복수형으로 이름을 짓는 것이 자연스럽습니다.

문서(Document) —상자 안에 들어가는 쪽지 한 장

"바다탐험가·740점·3-2반·10시 12분" 이 한 줄이 문서 하나입니다. 학생이 게임을 한 판 끝낼 때마다 쪽지 한 장이 상자에 들어갑니다.

필드(Field) —쪽지에 적는 항목

nickname ·score ·classCode ·createdAt 네 가지입니다. 여기에 이름이나 학번, 이메일은 넣지 않습니다. 무엇을 넣고 무엇을 넣지 않을지는 153쪽에서 한 번 더 못 박습니다.

실제로 저장되는 모습은 다음과 같습니다. scores 컬렉션 안의 문서 한 장을 펼쳐 본 것입니다.

```
{
   nickname:  "바다탐험가",
   score:     740,
   classCode: "3-2-r1",
   createdAt: 2026-08-25 10:12:07
}
```

"실시간"이 공짜가 아닌 이유

학생 화면 하나하나가 칠판을 계속 지켜보는 눈(리스너)을 하나씩 갖습니다. 누군가 점수를 올리면 지켜보던 모든 눈에게 바뀐 쪽지가 전달되고, 그 전달 하나하나가 "읽기"로 계산됩니다. 그래서 리더보드는 쓰기보다 읽기를 훨씬 많이 씁니다.

이 사실은 무료 한도를 계산할 때 결정적입니다. 쓰기가 아니라 읽기가 먼저 닳기 때문입니다. 170쪽에서 실제 사용량 화면과 함께 다시 다룹니다.

6.4 Firebase 콘솔에서 프로젝트 만들기

이제 콘솔을 엽니다. 주소창에 console.firebase.google.com을 치고 구글 계정으로 로그인합니다. 검색하지 말고 주소를 직접 치는 편이 빠르니, 즐겨찾기에 넣어 두세요. 프로젝트는 "우리 반 게임 데이터가 사는 집" 한 채이고, 만드는 데 세 단계가 걸립니다.

콘솔 첫 화면—지도부터 보기

로그인하면 아래 화면이 나옵니다. 왼쪽은 새로 만들기, 오른쪽은 이미 가진 프로젝트 목록입니다. 처음이라면 오른쪽이 비어 있습니다.

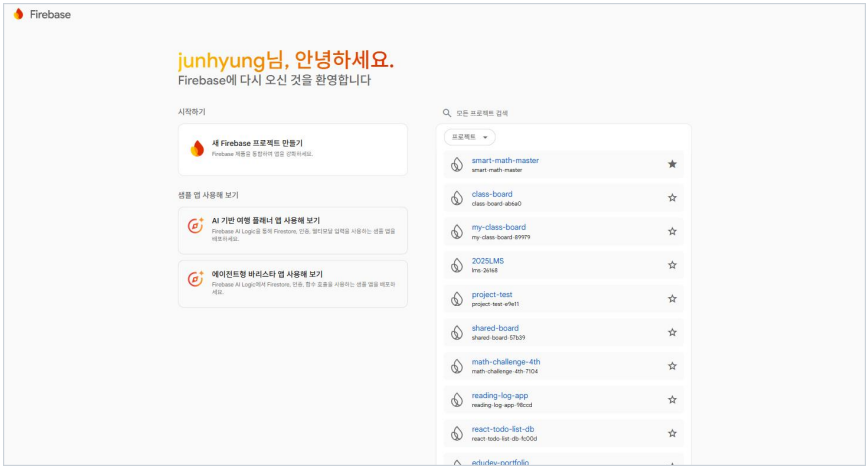


그림 6-1. Firebase 콘솔 홈. 왼쪽 "새 Firebase 프로젝트 만들기", 오른쪽은 이미 만든 프로젝트 목록

로그인은 교사 업무용 구글 계정으로 하기를 권합니다. 이 프로젝트는 만든 계정에 묶이기 때문입니다. 개인 계정으로 만들어 두면 나중에 인계하거나 학교를 옮길 때 곤란해집니다. 왼쪽 **새 Firebase 프로젝트 만들기**를 누르면 세 단계 마법사가 시작됩니다.

> **무료로 씁니다**
> Firebase의 **Spark 요금제**는 결제 카드 등록 없이 쓸 수 있고, 이 단원의 모든 실습이 그 안에서 끝납니다. 콘솔 왼쪽 아래에 "Spark 무료($0/월)"이 늘 표시되니, 진행 중에 한 번씩 확인하세요.

첫째 화면—이름 정하기

이름을 치면 그 아래에 자동으로 전 세계에서 유일한 프로젝트 ID가 붙습니다. 이 ID는 나중에 주소와 설정 곳곳에 나타나므로 알아보기 쉬운 영문 이름을 쓰는 편이 좋습니다.

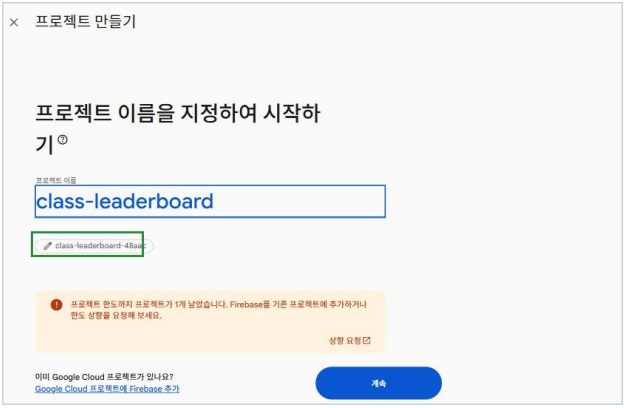


그림 6-2. 프로젝트 이름을 입력하면 아래에 프로젝트 ID가 자동으로 생성된다 (class-leaderboard-48aac)

1. **영문 이름으로 짓습니다.** class-leaderboard, math-4th-board처럼 씁니다. 한글도 되지만 ID가 복잡해집니다.
2. **프로젝트 ID는 자동으로 붙습니다.** 이름 뒤에 임의의 다섯 글자가 붙어 전 세계에서 유일해집니다. 연필 아이콘으로 고칠 수 있지만 그대로 두면 됩니다.
3. **"계속"을 누릅니다.** 아직 아무것도 만들어지지 않았습니다. 세 단계를 다 지나야 실제로 생성됩니다.

프로젝트 한도

한 구글 계정이 만들 수 있는 프로젝트 수에는 상한이 있습니다. 남은 개수가 적으면 화면에 안내 상자가 뜹니다. 한도에 걸렸다면 안 쓰는 프로젝트를 정리하거나, 기존 프로젝트에 Firestore만 추가해 쓰거나, 안내의 "상향 요청"을 누릅니다. 학년 전체가 각자 만든다면 연수 전에 미리 확인해 두세요.

둘째 화면—Firebase의 Gemini

두 번째 화면은 Firebase 콘솔 안에서 Gemini의 도움을 받을지 묻습니다. 콘솔에서 막힐 때 채팅으로 물어볼 수 있게 해 주는 기능이고, 기본은 켜짐입니다.

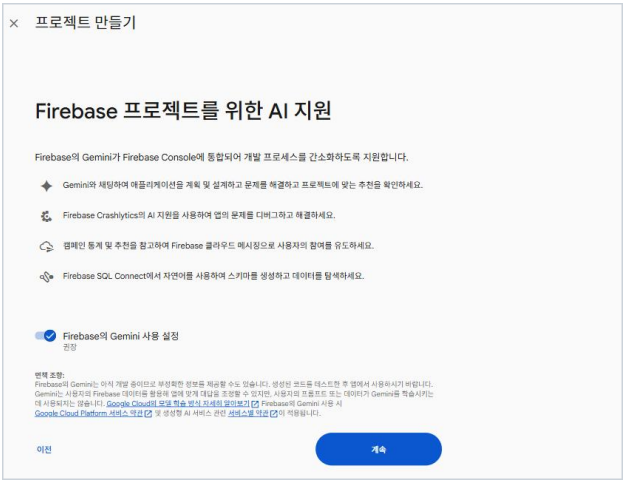


그림 6-3. "Firebase의 Gemini 사용 설정" 토글. 기본값은 켜짐

켜 두면 콘솔 화면에서 "이 규칙이 무슨 뜻이야?" 같은 질문을 바로 할 수 있습니다. Firestore 화면 위쪽에 뜨는 "Firestore를 사용하기 위한 핵심 개념에 관해 Gemini에게 물어보기" 버튼이 그것입니다. 다만 면책 조항에 적혀 있듯 생성된 답이 부정확할 수 있고, 교육청 지침상 AI 기능 사용에 제한이 있는 경우도 있습니다. 판단이 서지 않으면 꺼 두고 진행해도 이 교재의 모든 실습이 그대로 됩니다.

결정했으면 **계속**을 누릅니다. 아직도 프로젝트는 만들어지지 않았습니다. 다음 화면이 마지막 단계이고, 거기서 처음으로 실제 생성이 일어납니다.

셋째 화면—Google 애널리틱스는 끄세요

마지막 화면은 Google 애널리틱스를 켤지 묻고, 기본값은 켜짐입니다. 이 단원에서 기본값을 바꾸는 유일한 곳입니다.

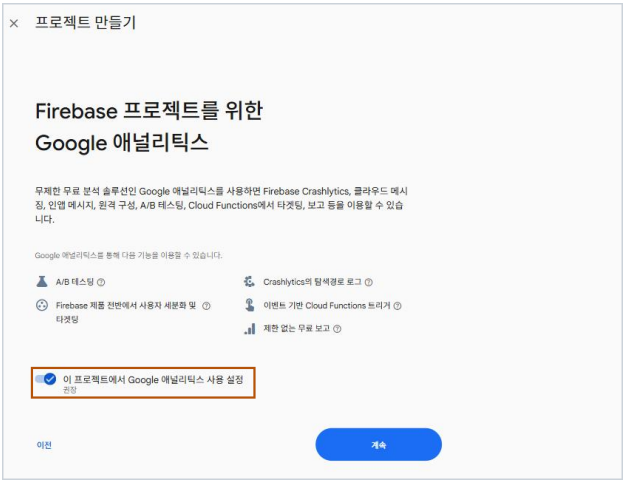


그림 6-4. Google 애널리틱스 화면. 기본값은 켜짐 상태다

애널리틱스는 누가 언제 어떤 화면을 봤는지 수집하고 분석하는 도구입니다. 우리가 만드는 것은 초등학생이 쓰는 학습 게임이고, 이 단원의 원칙은 "개인정보를 아예 수집하지 않는다"입니다. 그래서 이 토글은 끕니다. 토글을 끄면 아래 기능 목록에 취소선이 그어집니다.

그림 6-5. 토글을 끄면 기능 목록에 취소선이 붙는다. 이 상태가 되어야 꺼진 것

1. **토글을 클릭해 끕니다.** 기능 목록에 취소선이 그어지면 꺼진 것입니다.
2. **"프로젝트 만들기"를 누릅니다.** 버튼 이름이 "계속"에서 바뀌어 있습니다. 여기서 처음으로 실제 생성이 시작됩니다.
3. **30초에서 1분 기다립니다.** "준비 완료" 안내가 뜨면 계속 진행합니다.

> **왜 끄는가**
>
> 애널리틱스는 그 자체로 불법이 아닙니다. 다만 만 14세 미만 아동의 정보를 다루려면 법정대리인 동의가 필요하고(개인정보 보호법 제22조의2), 학생 이용 기록을 구글 계정과 연결해 쌓아 둘 이유가 우리 수업에는 없습니다. 켜지 않으면 동의 문제 자체가 생기지 않습니다. 나중에 필요해지면 콘솔에서 언제든 켤 수 있으니, 지금은 꺼 두는 쪽이 안전합니다.

프로젝트 개요 화면 읽는 법

프로젝트가 만들어지면 개요 화면으로 들어옵니다. 앞으로 이 화면을 계속 오가게 됩니다. 메뉴가 많아 보이지만 이 단원에서 쓰는 곳은 왼쪽 사이드바의 두 군데뿐입니다. Firestore(데이터)와 Authentication(인증). 나머지는 지금 몰라도 됩니다.

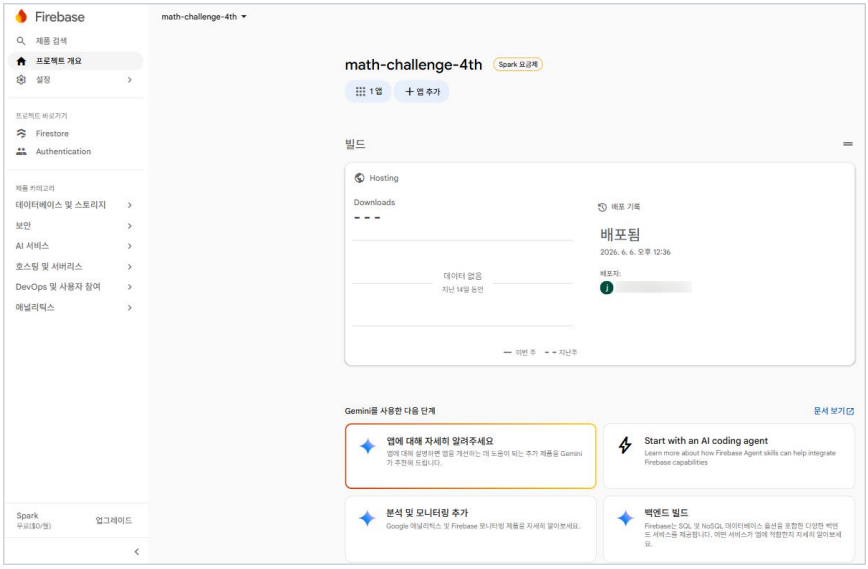


그림 6-6. 프로젝트 개요. 왼쪽 사이드바에 Firestore와 Authentication 바로가기가 있다

- **프로젝트 이름 옆 배지** —"Spark 요금제"는 무료 요금제라는 뜻입니다. 왼쪽 아래에도 "Spark 무료($0/월)"이 늘 표시됩니다.
- **프로젝트 바로가기** —Firestore와 Authentication. 이 단원에서 실제로 누르는 곳은 여기 둘뿐입니다.
- **제품 카테고리** —데이터베이스 및 스토리지, 보안, AI 서비스, 호스팅 등으로 묶여 있습니다. 바로가기에 없는 기능은 여기서 찾습니다.
- **앱 추가** —웹·안드로이드·iOS 앱을 이 프로젝트에 연결합니다. AI Studio를 쓰면 자동으로 붙으므로 직접 누를 일이 없습니다.
- **왼쪽 아래 업그레이드** —유료(Blaze) 전환 버튼입니다. 이 단원에서는 누르지 않습니다.

6.5 Firestore 열기—컬렉션·문서·필드 만들기

사이드바의 Firestore를 누릅니다. 처음이라면 "데이터베이스 만들기"부터 나오고, 위치와 보안 규칙 모드를 고르면 데이터 화면이 됩니다. 화면은 왼쪽부터 컬렉션 → 문서 → 필드 순서로 세 칸이 이어집니다. 위쪽 탭 여섯 개도 함께 봐 두세요.

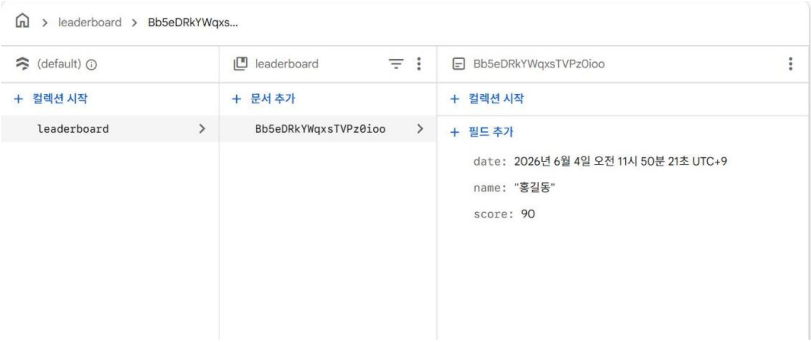


그림 6-7. Firestore 데이터 화면. 왼쪽부터 컬렉션 / 문서 / 필드 세 칸이 이어진다

3단 패널 구조 익히기

- **왼쪽 칸** —컬렉션 목록입니다. 그림에서는 leaderboard 하나가 있습니다.
- **가운데 칸** —선택한 컬렉션 안의 문서 목록입니다. 문서 ID는 자동 생성된 무작위 문자열입니다.
- **오른쪽 칸** —선택한 문서의 필드입니다. 그림에서는 date ·name ·score 세 개가 보입니다.
- **위쪽 탭** —데이터 / 규칙 / 색인 / 재해 복구 / 사용량 / 쿼리 통계. 우리는 앞의 둘과 사용량만 씁니다.
- **맨 아래** —"데이터베이스 위치: nam5"처럼 리전이 표시됩니다. 한 번 정하면 바꿀 수 없습니다.

컬렉션 만들기—상자에 이름 붙이기

왼쪽 칸 위의 **+ 컬렉션 시작**을 누르면 두 단계짜리 대화상자가 열립니다. 1단계는 상자 이름(컬렉션 ID)만 정하면 됩니다. 우리 리더보드는 점수를 담을 상자 하나면 충분하므로 scores라고 짓겠습니다.

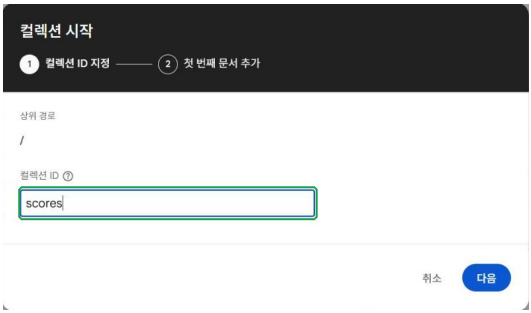


그림 6-8. 컬렉션 시작 1단계. 상위 경로 "/" 아래에 컬렉션 ID를 입력한다(초록 표시)

1. **왼쪽 칸의 "+ 컬렉션 시작".** 데이터 탭에서 첫 번째 칸 맨 위에 있습니다.
2. **상위 경로가 "/"인지 확인합니다.** 최상위에 만든다는 뜻입니다. 문서 안에 또 컬렉션을 넣을 수도 있지만 우리는 쓰지 않습니다.
3. **컬렉션 ID에 scores를 입력합니다.** 영문 소문자를 권합니다. 나중에 프롬프트와 보안 규칙에서 이 이름을 그대로 씁니다.
4. **"다음"을 누릅니다.** 컬렉션은 문서가 하나라도 있어야 만들어집니다. 그래서 2단계로 이어집니다.

> **이름 짓기 요령**
> 컬렉션은 여러 쪽지가 들어가는 상자이므로 복수형(scores, students)이 자연스럽습니다. 나중에 AI에게 지시할 때도 "scores 컬렉션에 저장해줘"처럼 그대로 부르게 됩니다.

첫 쪽지 한 장 써 보기

2단계는 상자에 넣을 첫 쪽지(문서)를 만듭니다. 여기서 문서 ID와 필드 이름, 값의 유형, 실제 값을 정합니다. 실제 수업에서는 학생이 게임을 하면 코드가 알아서 쪽지를 넣지만, 구조를 눈으로 익히려면 한 번은 직접 써 보는 편이 좋습니다.

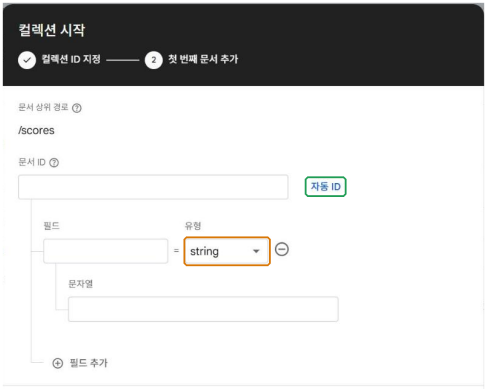


그림 6-9. 컬렉션 시작 2단계. 문서 ID는 자동 ID(초록 표시)로 두고, 필드와 유형(주황 표시), 값을 차례로 정한다

1. **문서 ID는 "자동 ID"를 클릭합니다.** 오른쪽의 **자동 ID**를 누르면 무작위 문자열이 채워집니다. 직접 짓지 마세요. 겹치면 서로 덮어씁니다.
2. **필드 이름에 nickname.** 왼쪽 "필드" 칸입니다. 영문 소문자로 짓고, 코드와 프롬프트, 규칙에서 같은 철자를 씁니다.
3. **유형은 string 그대로 둡니다.** 닉네임은 글자이므로 문자열입니다.
4. **값에 예시 닉네임을 입력합니다.** 그림에서는 "바다탐험가"입니다. 실명이 아니라 사물·동물·색깔 조합을 예시로 쓰세요.
5. **"필드 추가"로 나머지 세 개를 더합니다.** score(int64) · classCode(string) · createdAt(timestamp)를 같은 방식으로 넣습니다.
6. **"저장"을 누릅니다.** 컬렉션과 첫 문서가 함께 만들어집니다.

필드 유형—무엇을 골라야 하나

유형 칸을 누르면 목록이 펼쳐집니다. 이름은 낯설지만 교사가 실제로 쓰는 것은 셋뿐입니다. 유형을 잘못 고르면 순위가 엉뚱하게 정렬되므로, 이 대목은 한 번 정확히 짚고 갑니다.

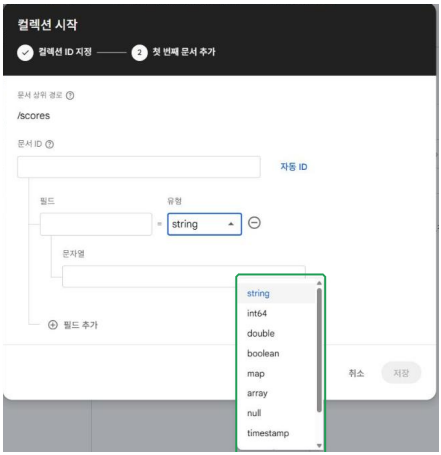


그림 6-10. 유형 목록(초록 표시). string · int64 · double · boolean · map · array · null · timestamp 순으로 나열된다

유형	무엇인가	우리 리더보드에서
string	글자. 닉네임, 반 코드처럼 글로 된 값	nickname, classCode
int64	정수. 소수점이 없는 숫자	score
timestamp	날짜와 시각	createdAt
double	소수. 평균 점수처럼 소수점이 있는 숫자	쓰지 않음
boolean	참/거짓. “클리어했는가” 같은 값	쓰지 않음
map / array	값 안에 값이 또 들어가는 형태	쓰지 않음
null	값이 비어 있음	쓰지 않음

> **score는 반드시 int64**
> 점수를 문자열(string)로 저장하면 "100"이 "90"보다 앞에 오는 사고가 납니다. 글자 순서로 정렬되기 때문입니다. string으로 두면 "1000점"이 "999점"보다 아래로 내려갑니다.

6.6 우리 리더보드의 데이터 설계

기능을 정하기 전에 "무엇을 저장할 것인가"를 먼저 정합니다. 여기서 정한 네 칸이 곧 프롬프트가 되고, 보안 규칙이 되고, 콘솔에서 눈으로 점검할 목록이 됩니다. 그리고 이 절에서는 넣을 것보다 넣지 않을 것이 더 중요합니다.

넣을 것—scores 컬렉션의 문서 한 장

필드	유형	담는 값
nickname	string	학생이 직접 정한 별명 (2–8자)
score	int64	그 판에서 얻은 점수
classCode	string	반과 회차 (예: 3-2-r1)
createdAt	timestamp	기록한 시각

넣지 않을 것—하나라도 들어가면 설계가 무너집니다

- **이름·학번** —실명은 물론 "3반민준이" 같은 별명도 실명입니다. 특정이 되는 순간 실명과 같습니다.
- **이메일·사진** —익명 로그인을 쓰면 애초에 받아오지 않습니다. 받아올 수 있다고 받아 두지 마세요.
- **학교 이름** —반 코드만으로 충분합니다. 학교명이 붙으면 특정이 쉬워집니다.
- **기기 정보·위치** —수집할 이유가 없습니다. 요청하지도 마세요.

이렇게 정해 두면 나중에 이 두 줄이 그대로 프롬프트가 됩니다. AI에게 리더보드를 맡기기 전에 이 문장을 먼저 넣어 두는 습관을 들이세요.

```
scores 컬렉션에 { nickname(문자), score(정수),
   classCode(문자), createdAt(시각) } 네 필드만 저장해줘.
이름·학번·이메일·사진·학교명은 절대 저장하지 마.
```

콘솔에서 눈으로 점검하기

수업을 한 번 돌린 뒤 콘솔의 Firestore 데이터 화면을 열어, 문서 하나를 골라 오른쪽 칸의 필드가 정확히 네 개뿐인지 확인하세요. 다섯 개째가 보인다면 AI가 무언가를 더 저장하고 있다는 뜻이고, 그것이 무엇인지 반드시 확인해야 합니다.

6.7 로그인을 걷어냅니다—초등 교실의 핵심 갈림길

Firebase 인증을 켜면 Google 로그인이 기본으로 따라옵니다. 편리하지만 초등 교실에서는 그대로 쓸 수 없습니다. 학생 계정으로 로그인하는 순간 이메일과 이름, 프로필 사진이 앱에 넘어오고, 만 14세 미만 아동의 개인정보 처리에는 법정대리인 동의가 필요해집니다.[2] 순위표에 필요한 것은 신원이 아니라 "누가 누구인지 구분만 되는 표식"입니다.

A안·익명 로그인과 닉네임 (권장)

개인정보를 하나도 받지 않으면서 사용자마다 임시 식별자(uid)를 발급합니다. 학생은 닉네임만 입력합니다. 이 단원의 실습은 이 길로 갑니다.

- 보안 규칙에서 "로그인한 사람만 쓰기 가능"을 걸 수 있어 장난 데이터를 막기 쉽습니다.
- 콘솔에서 익명 제공업체를 한 번 켜 줘야 합니다. 그 화면은 155쪽에서 안내합니다.
- 익명 계정은 30일 자동 정리 대상이 될 수 있으나, 회차마다 초기화하는 우리 설계에는 문제가 없습니다.

[2] 개인정보보호위원회. (2023). *개인정보 보호법 제22조의2.*; Firebase. (2026). *Authenticate with Firebase anonymously on the web.* Firebase Docs. https://firebase.google.com/docs/auth/web/anonymous-auth

B안·인증 없이 공개 쓰기 (차선)

로그인 자체를 없애고 누구나 점수를 남길 수 있게 규칙을 엽니다. 콘솔 설정이 필요 없어 가장 간단하지만, 대가가 큽니다.

- 주소만 알면 누구나 점수를 쓸 수 있습니다. 장난 점수(999999)를 막을 장치가 규칙 검증뿐입니다.
- 학교 밖에 주소가 퍼지면 손쓰기 어렵습니다.
- 한 차시짜리 공개 수업처럼 짧고 통제된 상황에서만 쓰세요.

A안으로 가기로 했다면, AI Studio에서 앱을 열고 다음 프롬프트를 넣습니다. 콘솔 작업과 앱 코드 작업이 짝을 이루어야 하므로, 이 프롬프트를 넣은 뒤 반드시 다음 절의 콘솔 설정까지 마쳐야 합니다.

```
실습 프롬프트 1 —로그인 걷어내고 닉네임으로

이 앱은 초등학생이 쓰는 앱이라 개인정보를 모으면 안 돼.
Google 로그인 화면을 없애고, 대신 Firebase 익명 로그인
(Anonymous Authentication)을 써줘. 게임 시작 전에 닉네임
입력창만 보여주고, 닉네임은 한글·영문 2--8자로 제한해줘.
이메일·이름·프로필 사진 같은 개인정보는 받지도 저장하지도 마.
```

6.8 익명 인증 켜기—콘솔에서 스위치 하나

사이드바에서 Authentication을 누릅니다. 처음이라면 "시작하기" 버튼이 보이고, 누르면 사용자 화면이 됩니다. 이 절의 실습은 클릭 네 번이면 끝나지만, 그 마지막 클릭을 빠뜨려서 실습이 멈추는 일이 가장 자주 일어납니다.

사용자 탭—지금 누가 등록되어 있나

첫 탭인 **사용자**는 이 앱에 등록된 계정 목록입니다. 그림에서는 구글 계정 하나가 등록되어 있습니다. 우리가 바꾸려는 것이 바로 이 상태입니다.

그림 6-11. Authentication 사용자 탭. 이메일과 UID는 흐림 처리했다

- **식별자** —누구인지 알아보는 값입니다. 구글 로그인이면 이메일 주소가 그대로 들어옵니다. 익명 로그인이면 이 칸이 비어 있습니다.
- **제공업체** —어떤 방식으로 로그인했는지 보여줍니다. 그림의 아이콘은 Google 로그인입니다.
- **사용자 UID** —Firebase가 내부적으로 쓰는 임의의 식별자입니다. 익명 로그인은 이것만 생기며, 개인정보가 아닙니다.
- **다섯 탭** —사용자 / 로그인 방법 / 템플릿 / 사용량 / 설정. 우리는 앞의 두 개만 씁니다.

익명 로그인으로 바꾸면 이 목록은 어떻게 달라질까요. 식별자 칸이 비고 제공업체 아이콘이 사라진 줄이 학생 수만큼 쌓입니다. 이름도 이메일도 없는 임의의 UID만 남습니다. 그것이 우리가 원하는 상태입니다.

로그인 방법 탭—제공업체 추가하기

두 번째 탭 **로그인 방법**으로 갑니다. 여기에 현재 켜져 있는 로그인 수단이 나열됩니다. 그림에서는 Google 하나가 "사용 설정됨" 상태입니다.

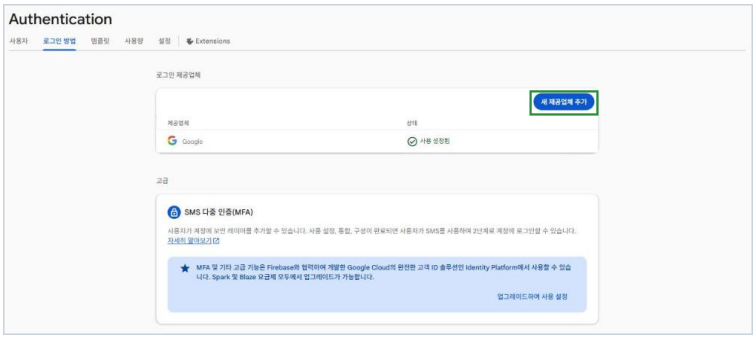


그림 6-12. 로그인 방법 탭. 제공업체 목록과 오른쪽 위 "새 제공업체 추가" 버튼

1. **Authentication → 로그인 방법 탭.** 제목 바로 아래 두 번째 탭입니다.
2. **현재 목록을 확인합니다.** Google이 "사용 설정됨"이면 그대로 두어도 됩니다. 익명을 더해도 서로 부딪히지 않습니다.
3. **"새 제공업체 추가"를 누릅니다.** 오른쪽 위의 파란 버튼입니다.

> **Google 로그인을 지울까요**
>
> 지우지 않아도 됩니다. 앱 코드에서 익명 로그인만 부르면 구글 로그인 화면은 나타나지 않습니다. 다만 학생용으로 배포하기 전에 실제로 구글 로그인 화면이 뜨지 않는지 반드시 한 번 확인하세요.

제공업체 목록에서 “익명” 고르기

대화상자가 열리면 로그인 수단이 세 묶음으로 나뉘어 보입니다. 우리가 찾는 것은 왼쪽 **기본 제공업체** 맨 아래의 **익명**입니다. 이름 그대로, 아무 정보도 받지 않고 사용자를 구분만 해 주는 방식입니다.

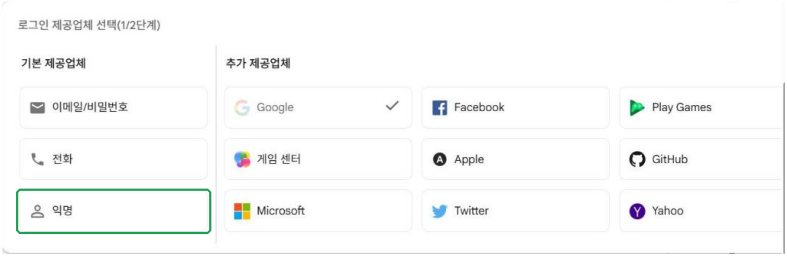


그림 6-13. 로그인 제공업체 선택(1/2단계). 왼쪽 기본 제공업체 맨 아래가 익명이다(초록 표시)

- **기본 제공업체** —이메일/비밀번호, 전화, 익명. 우리가 쓰는 것은 익명 하나입니다.
- **추가 제공업체** —Google, Facebook, Apple, GitHub 등 외부 계정 로그인입니다. 초등 수업에서는 쓰지 않습니다.
- **커스텀 제공업체** —학교나 기관의 자체 인증 시스템을 붙일 때 씁니다. 유료 요금제가 필요합니다.

> **“이메일/비밀번호”는 왜 안 되나**
>
> 초등학생에게 계정과 비밀번호를 만들게 하는 순간 관리 책임이 교사에게 넘어옵니다. 잊어버린 비밀번호를 재설정해 주는 일이 매 수업 생깁니다. 우리 목적(순위 구분)에는 익명이면 충분합니다.

사용 설정—이 스위치 하나가 실습을 좌우합니다

익명을 고르면 2/2단계가 나옵니다. 할 일은 스위치를 켜고 저장하는 것뿐입니다.

그림 6-14. 제공업체 구성(2/2단계). 오른쪽 위 "사용 설정" 토글(초록 표시)과 아래 "자동 정리" 체크박스

1. **"사용 설정" 토글을 켭니다.** 오른쪽 위 스위치입니다. 그림은 꺼진 상태이고, 켜면 파란색이 됩니다.
2. **"자동 정리"도 켜기를 권합니다.** 30일이 지난 익명 계정을 자동 삭제합니다. 회차마다 초기화하는 우리 설계에서는 잃을 것이 없고, 사용량·결제 할당량에도 집계되지 않습니다.
3. **"저장"을 누릅니다.** 목록에 "익명·사용 설정됨"이 추가되면 완료입니다.

> **켜지 않으면 어떻게 되나**
>
> 앱은 정상으로 보입니다. 게임도 잘 돌아가고 오류 메시지도 뜨지 않습니다. 그런데 점수만 저장되지 않고, 순위표는 계속 비어 있습니다. 원인을 찾느라 프롬프트를 몇 번이고 다시 쓰게 되는 자리입니다.
>
> 실습 중 "순위표가 비어요"라는 말이 나오면 먼저 이 스위치부터 확인하세요. 절반 이상이 여기서 해결됩니다.

6.9 보안 규칙—지금 내 데이터베이스는 안전한가

Firestore의 **규칙** 탭을 엽니다. 여기에 쓰인 몇 줄이 데이터베이스 앞을 지키는 문지기 전부입니다. 아래는 실제 교사 프로젝트에서 촬영한 화면입니다. 다섯 번째 줄을 소리 내어 읽어 보세요. 이 단원에서 가장 중요한 화면입니다.

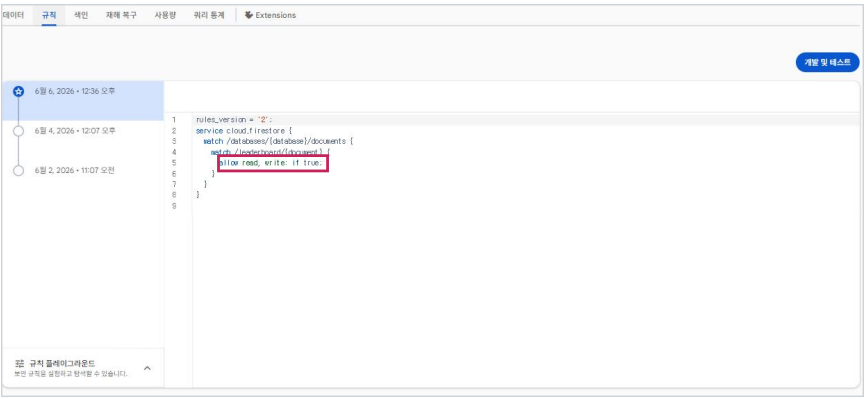


그림 6-15. 규칙 탭. 왼쪽은 버전 기록, 오른쪽이 현재 적용 중인 규칙이다

지금 적용 중인 규칙은 이렇습니다.

```
match /leaderboard/{document} {
   allow read, write: if true;
}
```

이것이 뜻하는 것은 하나입니다. "조건 없이(if true) 읽기와 쓰기를 모두 허용한다." 즉 주소만 알면 누구나 점수를 넣고, 남의 점수를 고치고, 전부 지울 수 있습니다.

드문 일이 아닙니다. 테스트 모드로 시작하거나 AI가 초안을 쓰면 흔히 이 상태로 남습니다. 학생에게 열기 전에 반드시 고쳐야 합니다.

규칙 문장 해부—여섯 줄만 읽으면 됩니다

보안 규칙은 프로그래밍 언어처럼 보이지만 구조는 아주 단순합니다. "어느 경로에(match) 어떤 동작을(allow) 어떤 조건일 때(if) 허용한다" 한 문장의 반복입니다. 규칙 탭에 실제로 들어 있는 전체 문장은 다음과 같습니다.

```
rules_version = '2';
service cloud.firestore {
   match /databases/{database}/documents {
      match /leaderboard/{document} {
         allow read, write: if true;
      }
   }
}
```

여섯 개의 뜻만 알면 내 규칙이 안전한지 스스로 판단할 수 있습니다. 앞의 셋은 고정이고, 뒤의 셋이 교사가 실제로 손대는 부분입니다.

1. **rules_version** —규칙 문법의 버전입니다. '2'로 두면 됩니다.
2. **service cloud.firestore** —이 규칙이 Firestore용이라는 선언입니다. 고정입니다.
3. **match /…/documents** —이 데이터베이스의 모든 문서를 뜻하는 바깥 울타리입니다. 고정입니다.
4. **match /leaderboard/{…}** —어느 상자에 적용할지 정합니다. leaderboard 컬렉션의 모든 문서라는 뜻입니다. 우리 실습에서는 이 자리가 scores로 바뀝니다.
5. **allow read, write** —무엇을 허용할지 정합니다. read는 읽기, write는 만들기·고치기·지우기 전부입니다.
6. **if true** —언제 허용할지 정합니다. true는 언제나, 즉 조건이 없다는 뜻입니다. 가장 위험한 값입니다.

> **write는 셋을 합친 말입니다**
>
> write는 create(새로 만들기) ·update(고치기) ·delete(지우기)를 합친 말입니다. 그래서 "allow write"라고 쓰면 남의 점수를 고치고 지우는 것까지 함께 열립니다. 우리는 다음 절에서 이 셋을 갈라, create만 허용하고 update와 delete는 막을 것입니다.
>
> 반면 read를 열어 두는 것은 괜찮습니다. 순위표는 모두가 봐야 하니까요. 다만 그

> 것은 저장하는 필드에 개인정보가 없기 때문에 괜찮은 것입니다(153쪽 설계).

6.10 규칙 고쳐 쓰고 시험해 보기

규칙도 프롬프트로 고칩니다. 직접 타이핑할 필요가 없습니다. 다만 "점수는 몇 점까지가 정상인가" 같은 것은 우리 교실 사정이라 AI가 알 수 없습니다. 그 부분만 문장으로 적어 주면 됩니다.

프롬프트로 문지기 손보기

AI Studio에서 앱을 열고 다음 프롬프트를 넣습니다. 결과로 나온 규칙을 콘솔의 규칙 탭에 붙여넣고 **게시**를 누르면 반영됩니다.

```
실습 프롬프트 2 —문지기 손보기

firestore.rules를 이렇게 고쳐줘.
·scores 컬렉션은 로그인(익명 포함)한 사용자만 create 가능
·score는 정수이고 0 이상 99999 이하일 때만 허용
·nickname은 문자열이고 8자 이하일 때만 허용
·이미 저장된 문서는 update·delete 모두 거부
·읽기(read)는 누구나 가능
```

이런 규칙이 나옵니다. 아래는 실제로 돌아온 결과의 예시입니다.

```
match /scores/{doc} {
   allow read: if true;
   allow create: if request.auth != null
      && request.resource.data.score is int
      && request.resource.data.score >= 0
      && request.resource.data.score <= 99999
      && request.resource.data.nickname is string
      && request.resource.data.nickname.size() <= 8;
   allow update, delete: if false;
}
```

한 줄씩 읽어 보면 앞 절에서 익힌 구조 그대로입니다.

1. **allow read: if true** —읽기는 조건 없이 허용합니다. 순위표는 모두가 봐야 하고, 저장된 필드에 개인정보가 없으므로 이 줄은 열어 둡니다.
2. **allow create: if request.auth != null** —만들기는 로그인한 사용자만 허용합니다. 익명 로그인도 로그인이므로 여기에 걸립니다. 로그인하지 않은 요청은 이 줄에서 거부됩니다.
3. **score is int / >= 0 / <= 99999** —점수가 정수이고 0 이상 99999 이하일 때만 통과시킵니다. 세 조건이 모두 참이어야 합니다.
4. **nickname is string / size() <= 8** —닉네임이 문자열이고 여덟 자 이하일 때만 통과시킵니다. 긴 문장을 닉네임 칸에 밀어 넣는 장난을 여기서 막습니다.
5. **allow update, delete: if false** —고치기와 지우기는 언제나 거부합니다. 조건이 false이므로 어떤 경우에도 통과하지 않습니다.

왜 점수 상한을 두고 수정·삭제를 막나

개발자 도구로 값을 바꿔 999999점을 올리는 학생이 실제로 나옵니다. 우리 게임의 최고점보다 조금 넉넉한 값을 상한으로 잡아 두면, 그런 시도가 규칙 단계에서 걸립니다. 상한 숫자는 우리 게임에 맞게 고쳐 쓰세요.

수정과 삭제를 막는 이유는 다릅니다. 한 번 올린 점수를 누가 고치거나 지우면 순위표를 믿을 수 없습니다. 지우는 일은 교사가 콘솔에서만 하도록 남깁니다. 보안 규칙은 앱에서 오는 요청만 막을 뿐, 프로젝트 주인인 교사의 콘솔 작업은 막지 않기 때문입니다.

> **붙여넣은 뒤 반드시 "게시"**
>
> 편집만 하고 게시하지 않으면 예전 규칙이 계속 살아 있습니다. 화면에는 새 규칙이 보이는데 실제로는 옛 규칙이 도는 상태가 되어, 원인을 찾기가 대단히 어렵습니다. 왼쪽 버전 기록에 새 날짜가 생기면 반영된 것입니다.

규칙 플레이그라운드—게시하기 전에 시험해 보기

규칙은 조용히 실패합니다. 화면에 오류가 안 뜬다고 통과한 것이 아닙니다. 그래서 콘솔에는 규칙을 실제로 시험해 보는 도구가 있습니다. 규칙 탭 왼쪽 아래의 **규칙 플레이그라운드**를 펼치면 됩니다.

그림 6-16. 규칙 플레이그라운드. 시뮬레이션 유형·위치·"인증됨" 토글·실행 버튼

1. **시뮬레이션 유형** —get(읽기) / create(만들기) / update(고치기) / delete(지우기) 중 시험할 동작을 고릅니다.
2. **위치** —경로를 적습니다. 예를 들어 scores/test1입니다.
3. **"인증됨" 토글** —켜면 로그인한 사용자로, 끄면 로그인하지 않은 사람으로 시험합니다.
4. **"실행"을 누릅니다.** 허용과 거부가 색으로 표시되고, 규칙의 어느 줄에서 걸렸는지도 알려 줍니다.

꼭 해 볼 네 가지

인증됨을 켜고 create를 실행하면 허용되어야 합니다. 인증됨을 끄고 create를 실행하면 거부되어야 합니다. 인증됨을 켜고 update를 실행하면 거부되어야 합니다. 인증됨을 끄고 get을 실행하면 허용되어야 합니다. 순위표는 모두가 보기 때문입니다.

네 가지가 모두 예상대로 나오면 규칙이 제대로 걸린 것입니다. 하나라도 어긋나면 게시하지 말고 규칙을 다시 고치세요.

6.11 앱에 붙이기—Enable Firebase

여기서 무대가 Firebase 콘솔에서 AI Studio로 옮겨 갑니다. 5단원에서 쓰던 앱을 열고 오른쪽 위 톱니바퀴를 눌러 패널을 엽니다. 탭 중 Integrations를 고르고 검색창에 "fire"를 치면 하나가 나옵니다. Firebase Firestore & Auth입니다.

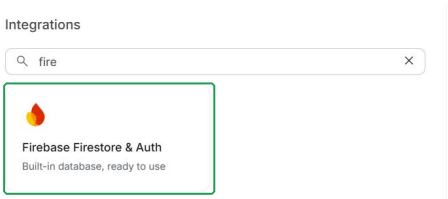


그림 6-17. Integrations 탭에서 "fire"로 검색한 결과. Firebase Firestore & Auth 카드(초록 표시)만 남는다

1. **5단원에서 만든 앱을 엽니다.** aistudio.google.com/apps → My apps에서 고릅니다.
2. **오른쪽 위 톱니바퀴를 클릭합니다.** 패널이 열리고 Chat / Share / Publish / Versions / GitHub / Integrations / Secrets 탭이 보입니다.
3. **Integrations 탭을 선택합니다.** 구글 앱 연동 목록이 나옵니다. 드라이브·시트·지메일 등이 함께 보입니다.
4. **검색창에 fire를 입력합니다.** "Firebase Firestore & Auth —Built-in database, ready to use" 카드가 남습니다.
5. **카드를 클릭합니다.** 설정 패널로 넘어갑니다.

> 프롬프트에 "Firebase로 저장해줘"라고 쓰면 이 카드가 자동으로 뜨기도 합니다. 둘 중 어느 쪽이든 도착지는 같습니다.

Enable Firebase 패널 뜯어보기

카드를 누르면 설정 패널이 열립니다. 맨 아래 Enable Firebase 한 번이면 데이터베이스와 인증, 보안 규칙 초안이 자동으로 붙습니다. 다만 그 위의 Your data will be stored in은 한 번 정하면 바꿀 수 없습니다. 누르기 전에 여기부터 보세요.

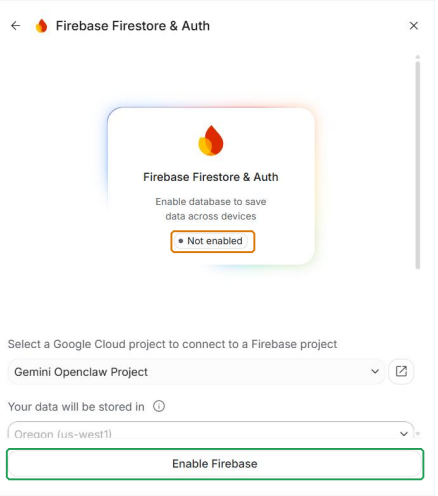


그림 6-18. Enable Firebase 패널. 상태(주황 표시) · 프로젝트 · 데이터 저장 위치, 맨 아래 Enable Firebase(초록 표시)

- **상태 표시** —카드에 "Not enabled"라고 적혀 있으면 아직 붙지 않은 상태입니다.
- **Google Cloud 프로젝트** —어느 프로젝트에 붙일지 고릅니다. 앞에서 만든 프로젝트를 선택하세요.
- **데이터 저장 위치** —기본값은 Oregon(us-west1), 즉 미국입니다. 국내 asia-northeast3(서울)로 바꾸는 편이 좋습니다. 한 번 정하면 변경할 수 없습니다.
- **App support email** —문제가 생겼을 때 표시될 연락처입니다. 교사 업무용 계정으로 둡니다.
- **Database ID** —자동으로 붙는 이름입니다. 손대지 않습니다.
- **Included services** —Cloud Firestore(데이터 저장·동기화)와 Firebase Authentication(로그인·접근 제어)이 함께 켜집니다.
- **Enable Firebase** —누르면 실제로 붙습니다. 잠시 뒤 코드에 firebase 설정 파일과 firestore.rules가 생깁니다.

firebaseConfig —이 값은 비밀이 아닙니다

연동이 끝나면 코드에 firebaseConfig라는 덩어리가 생깁니다. 콘솔의 프로젝트 설정에서도 같은 값을 볼 수 있습니다. apiKey라는 이름 때문에 5단원에서 배운 "키를 공개 저장소에 올리지 마라"와 충돌하는 것처럼 보이지만, 성격이 다릅니다.

그림 6-19. 프로젝트 설정 → 일반 → 내 앱의 SDK 설정. 실제 값은 흐림 처리했다

Gemini API 키는 진짜 비밀입니다. 이 키를 가진 사람은 내 계정으로 AI를 호출해 요금을 발생시킬 수 있습니다. 그래서 공개 저장소에 올리면 안 되고, 서버 쪽에 숨겨야 합니다. 5단원에서 경계한 것이 이쪽입니다.

firebaseConfig의 apiKey는 비밀이 아닙니다. "우리 프로젝트가 어디에 있는지"를 가리키는 주소표에 가깝습니다. 누구나 브라우저 개발자 도구로 볼 수 있고, 그렇게 설계되어 있습니다. 실제 방어는 보안 규칙이 합니다.

> **한 줄 요약**
> 문은 열려 있어도 됩니다. 문 안쪽에서 문지기(보안 규칙)가 지키니까요. 그래서 160쪽의 규칙 점검이 이 단원에서 가장 중요한 절인 것입니다.

6.12 리더보드 구현—원칙을 프롬프트로 옮기기

여기서 139쪽에서 정한 다섯 가지 원칙이 실제 문장이 됩니다. "순위표 만들어줘"라고만 하면 AI는 전체 순위를 실명 비슷하게 주르륵 나열하는 흔한 형태로 만듭니다. 원칙을 문장으로 적어 넣는 것이 이 단원의 실력입니다.

```
실습 프롬프트 3 —리더보드 구현

리더보드를 이렇게 만들어줘.
·scores 컬렉션에 { nickname, score, classCode, createdAt }
  문서를 저장
·화면 오른쪽에 TOP 10만 실시간으로 표시 (1--3위는 금·은·동 색)
·전체 순위는 보여주지 마. 내가 10위 밖이면 목록 아래에
  "나: 17위 / 340점" 한 줄만 내 화면에 따로 보여줘
·TOP 10 옆에"내 최고 기록"을 항상 같이 띄워줘
·같은 닉네임은 최고 점수 하나만 남겨줘
·classCode는 주소 뒤?class=3-2 로 받아 반별로 순위를 나눠줘
```

프롬프트의 각 줄이 어떤 원칙에서 나왔는가

원칙	프롬프트의 장치	확인 방법
실명 금지	nickname 필드만 저장. 이름·학번 필드를 아예 만들지 않음	콘솔에서 필드가 네 개뿐인지 확인
회차 초기화	classCode에 회차를 붙여 판을 나눔	주소 뒤 class 값이 바뀌는지 확인
내 기록 병기	“내 최고 기록”을 TOP 10 옆에 상설 배치	실행 화면 오른쪽에 늘 보이는지
TOP 10까지만	전체 순위 금지, 내 순위는 내 화면에만	11위 이하가 목록에 없는지
점수는 성취가 아님	최고 점수 하나만 남겨 반복 누적을 막음	같은 닉네임 줄이 하나뿐인지

회차 초기화는 이 프롬프트에서 반쪽만 구현됩니다. 나머지 반쪽인 교사용 패널은 다음 절에서 만듭니다.

> **실행 화면에서 확인할 것**
>
> TOP 10만 나오는가. 내 최고 기록이 옆에 있는가. 10위 밖일 때 내 순위 한 줄이 뜨는가. 옆자리 기기에서 점수를 올리면 새로고침 없이 바뀌는가.
>
> 그리고 Firebase 콘솔의 Firestore 데이터 화면을 열어, 문서에 필드가 정확히 네 개뿐인지 눈으로 확인하세요. 화면만 보고 넘어가면 무엇이 저장되고 있는지 끝까지 모릅니다.

6.13 교사용 회차 초기화—지우지 않고 새 판을 엽니다

139쪽 둘째 원칙(회차마다 초기화)을 구현할 차례입니다. 그런데 방금 우리는 규칙에서 삭제를 막았습니다. 모순처럼 보이지만 아닙니다. 초기화를 "지우기"가 아니라 "새 회차 코드 발급"으로 설계하면, 기록은 안전하게 남고 화면은 깨끗하게 비워집니다.

```
실습 프롬프트 4 —교사용 회차 패널

교사용 패널을 만들어줘.
·주소 뒤에?teacher=1 을 붙였을 때만 보이는 작은 패널
·"새 회차 시작" 버튼: classCode 뒤에 회차 번호를 붙여
  새 코드를 만들고(예: 3-2-r2) 그 코드로 순위표를 새로 시작
·이전 회차 기록은 지우지 말고 그대로 남겨둬
·현재 회차의 학생용 링크와 QR을 패널에 보여줘
```

회차 코드가 도는 방식

시점	코드	무슨 일이 일어나나
1교시	3-2-r1	1반 1회차. 학생 링크는 주소 뒤?class=3-2-r1
다음 차시	3-2-r2	"새 회차 시작"을 누르면 화면이 비워지고 새 링크가 발급됨
옆 반	3-3-r1	반이 다르면 코드가 달라 순위가 섞이지 않음

지난 회차의 기록은 데이터베이스에 그대로 남아 있습니다. 화면에서 사라진 것은 지금 회차 코드와 다른 기록을 보여주지 않기 때문일 뿐입니다. 학년 말에 "우리 반이 한 학기 동안 몇 판을 했는가"를 세어 보고 싶다면 그 자료가 그대로 있습니다.

진짜로 지워야 할 때

실수로 부적절한 닉네임이 올라갔거나 학년 말에 정리할 때는 Firebase 콘솔에서 직접 지웁니다. 콘솔 → Firestore → 문서를 고른 뒤 오른쪽 위 점 세 개 메뉴에서 삭제하면 됩니다. 보안 규칙은 앱에서 오는 요청만 막을 뿐, 프로젝트 주인인 교사

의 콘솔 작업은 막지 않습니다.

즉 삭제 권한을 교사 한 사람에게만 남겨 두는 구조입니다. 이것이 규칙에서 delete를 거부한 진짜 이유입니다.

> **?teacher=1 은 잠금장치가 아닙니다**
>
> 주소를 알아낸 학생은 똑같이 열 수 있습니다. 이것은 "평소에는 화면을 어지럽히지 않기 위한 정리 장치"이지 보안이 아닙니다. 진짜로 막아야 할 것—점수 조작과 삭제—은 160쪽과 162쪽의 보안 규칙이 막습니다.

6.14 사용량과 무료 한도—우리 학교는 몇 반까지

Firestore의 **사용량** 탭을 열면 이 데이터베이스가 얼마나 쓰였는지 그래프로 보입니다. 수업 한 차시를 돌린 다음 이 화면을 한 번 열어 보세요. 141쪽에서 말한 "읽기가 쓰기보다 훨씬 많다"를 직접 확인할 수 있습니다.

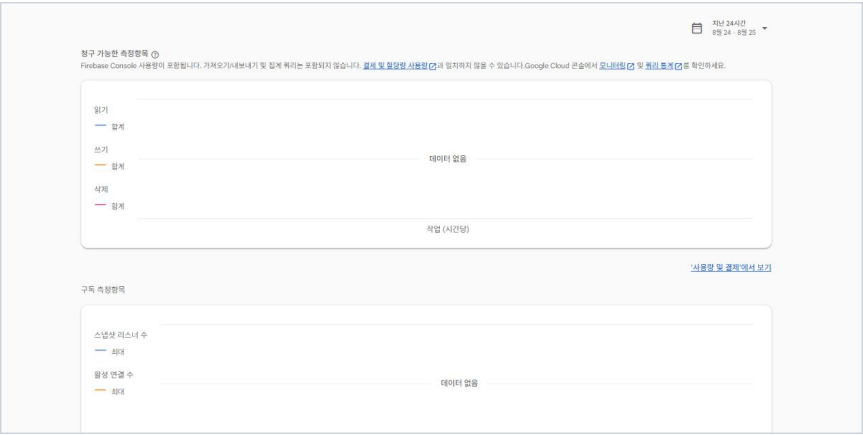


그림 6-20. 사용량 탭. 청구 가능한 측정항목(읽기·쓰기·삭제)과 구독 측정항목(스냅샷 리스너·활성 연결)

- **읽기** —순위표가 갱신되어 화면에 전달될 때마다 1입니다. 가장 먼저 닳습니다.

- **쓰기** —학생이 점수를 저장할 때마다 1입니다. 생각보다 적게 씁니다.
- **삭제** —우리 규칙은 삭제를 막았으므로 거의 0이어야 정상입니다. 0이 아니라면 규칙을 다시 확인하세요.
- **스냅샷 리스너 수** —지금 순위표를 지켜보고 있는 화면의 개수입니다. 접속한 학생 수에 가깝습니다.
- **활성 연결 수** —동시에 붙어 있는 연결입니다. 수업 중 실시간 참여 인원을 가늠할 수 있습니다.

공식 공유 한도

AI Studio가 만들어 준 데이터베이스는 결제 계정 없이 쓰는 대신 하루 사용량 한도를 여러 앱이 나눠 씁니다. 한도를 넘으면 태평양 시간 자정 무렵 다음 날에 초기화됩니다.

항목	한도	설명
저장 용량	1 GiB	점수 한 건이 100바이트도 안 되니 사실상 걸릴 일 없음
쓰기(write)	40,000 / 일	점수를 저장할 때마다 1
읽기(read)	50,000 / 일	실시간 갱신이 여기를 씀. 가장 먼저 닳는 칸
실시간 업데이트	50,000 / 일	지켜보던 화면에 변경이 전달될 때마다 1
네트워크 송신	10 GiB / 월	텍스트만 오가므로 여유 있음

한 반 40분 수업 추정

이 한도를 교실 규모로 옮겨 보면 다음과 같습니다. 직접 계산해 보시기를 권합니다.

```
학생 30명 × 한 명당 20회 도전
쓰기 = 600건 (한도의 1.5%)
읽기 = 약 15,000--20,000건 (한도의 30--40%)
하루 2--3개 학급이 안전선
```

읽기를 아끼는 방법은 두 가지입니다. 첫째, TOP 10만 구독합니다. 전체 순위를

실시간으로 받으면 읽기가 몇 배로 늡니다. 프롬프트에서 TOP 10만 표시하라고 못 박은 데에는 교육적 이유와 함께 이 실용적 이유도 있습니다. 둘째, 학년 전체를 하루에 돌린다면 실시간 대신 "게임이 끝날 때만 갱신"으로 바꿔 달라고 요청하세요.

한도를 넘겼더라도 앱이 멈추지는 않습니다. 저장과 갱신만 실패하고, 다음 날 자정 무렵 자동으로 풀립니다.

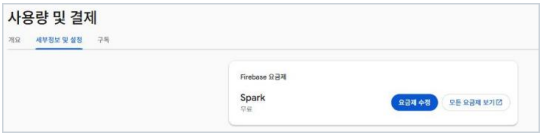


그림 6-21. 사용량 및 결제 → 세부정보 및 설정에서 현재 요금제를 확인할 수 있다. Spark가 무료 요금제다

6.15 학생에게 열기 전 여섯 칸

5단원의 여섯 칸(학생 이름·사진·API 키·저작권·닉네임·회수 절차)은 그대로 유효합니다. 여기에 데이터베이스가 붙으면서 새로 생긴 위험이 여섯 가지 더 있습니다. 첫 수업 전에 소리 내어 읽고 체크하세요.

- ☐ **보안 규칙 확인** —if true로 열려 있지 않은지 규칙 탭에서 직접 눈으로 확인합니다. 이 단원에서 가장 중요한 한 칸입니다(160쪽).
- ☐ **닉네임 필터** —학생은 부적절한 닉네임을 넣습니다. 반드시 넣습니다. 금지어 목록을 프롬프트에 넣고, 뚫리면 콘솔에서 지웁니다.
- ☐ **실명과 별명 주의** —"김민준"은 물론 "3반민준이"도 실명입니다. 사물·동물·색깔 조합을 예시로 주세요.
- ☐ **리전은 변경 불가** —Enable Firebase 전에 위치를 정해야 합니다. 국내 asia-northeast3(서울)을 권장합니다(165쪽).
- ☐ **교사 계정으로만** —Firebase 프로젝트는 만든 계정에 묶입니다. 업무용 계정으로 만들고, 학생에게 콘솔 주소를 알려주지 않습니다.
- ☐ **수업 후 처리** —학년이 끝나면 프로젝트를 정리하거나 데이터를 삭제합니다. 익명 데이터라도 남길 이유가 없으면 지우는 편이 낫습니다.

되짚기

우리 설계는 이메일도 이름도 받지 않습니다(익명 로그인과 닉네임). 그래서 만 14세 미만 아동의 개인정보 처리에 해당하지 않도록 만든 것입니다. 이 전제가 무너지는 순간—로그인을 다시 켜거나, 실명 입력칸을 추가하는 순간—법정대리인 동의 문제가 되살아납니다.

6.16 100분 미션—우리 반 순위표 완성하기

5단원에서 배포한 게임을 그대로 재료로 씁니다. 그림을 옆에 두고 순서대로 따라오세요. 마지막 산출물은 "옆 선생님 기기에서 내 순위표가 실시간으로 바뀌는 장면"입니다. 연수에서는 이 100분을 두 차시로 나누어 운영하기를 권합니다.

0–8분 **설계 먼저 정하기**
다섯 가지 원칙 중 우리 반에 그대로 적용할 것과 바꿀 것을 표시하고, 옆 선생님과 이유를 나눕니다(139쪽).

8–20분 **프로젝트 만들기**
콘솔 접속 → 이름 → Gemini 선택 → 애널리틱스 끄기 → 프로젝트 만들기(143쪽).

20–35분 **Firestore 열기**
컬렉션 scores를 만들고 첫 문서를 직접 입력한 뒤 필드 유형을 확인합니다(149쪽).

35–47분 **익명 인증 켜기**
Authentication → 로그인 방법 → 새 제공업체 추가 → 익명 → 사용 설정 → 저장(155쪽).

47–65분 **보안 규칙 검사**
규칙 탭에서 현재 규칙을 확인하고 프롬프트 2로 고친 뒤 게시하고, 플레이그라운드에서 네 가지를 시험합니다(162쪽).

65–75분 **앱에 붙이기**
AI Studio → Integrations → fire 검색 → 리전을 서울로 → Enable Firebase(165쪽).

75 – 90분 **리더보드 구현**
프롬프트 3을 넣고 TOP 10 ·내 최고 기록·내 순위 세 가지를 확인한 뒤, 콘솔에서 필드가 네 개인지 확인합니다(167쪽).

90 – 100분 **회차 패널과 공유회**
프롬프트 4를 넣고 Publish로 배포한 뒤, 옆 선생님 기기에서 접속해 동시에 점수를 올려 봅니다(169쪽).

> **제출 양식**
> 교과·학년: _____
> 프로젝트 ID: _____
> 리전: _____ 회차 코드: _____
> 배포 주소(Cloud Run): _____
> 규칙 시험 4종 통과: □□□□ 동시 접속 확인: □

6.17 자주 막히는 지점 여덟

리더보드 실습에서 가장 많이 나오는 질문입니다. 절반은 "콘솔에서 스위치 하나를 안 켰다"에 해당합니다.

Q. 점수가 저장이 안 돼요

A. 가장 흔한 원인은 익명 로그인을 켜지 않은 것입니다. Authentication → 로그인 방법 → 익명 → 사용 설정을 확인하세요(155쪽). 그래도 안 되면 보안 규칙을 봅니다.

Q. 오류는 없는데 순위표가 비어요

A. 규칙은 조용히 거부합니다. 콘솔의 Firestore에서 문서가 실제로 쌓이는지 직접 확인하는 것이 가장 빠릅니다(149쪽).

Q. 프로젝트를 더 못 만든대요

A. 계정당 프로젝트 한도입니다. 안 쓰는 프로젝트를 정리하거나, 기존 프로젝트에 Firestore만 추가해 쓰세요(143쪽).

Q. 리전을 잘못 골랐어요

A. 위치는 한 번 정하면 바꿀 수 없습니다. 실습 단계라면 새 프로젝트로 다시 만드는 편이 빠릅니다(165쪽).

Q. 1000점이 999점보다 아래예요

A. score를 string으로 저장했기 때문입니다. 글자 순서로 정렬됩니다. int64로 바꿔 달라고 요청하세요(149쪽).

Q. GitHub Pages에서 순위표가 안 보여요

A. 정상입니다. 정적 호스팅에는 서버가 없습니다. 리더보드 버전은 Publish로 만든 Cloud Run 주소를 학생에게 나눠 주세요(140쪽).

Q. 이상한 닉네임이 올라왔어요

A. 콘솔 Firestore에서 해당 문서를 삭제합니다. 앱에서는 지울 수 없게 막아 두었으므로 교사만 지울 수 있습니다(169쪽).

Q. 규칙을 고쳤는데 그대로예요

A. "게시"를 누르지 않았을 가능성이 큽니다. 편집만으로는 반영되지 않습니다. 왼쪽 버전 기록에 새 날짜가 생겼는지 확인하세요(162쪽).

6.18 단원 마무리

이제 우리 반의 기록이 한곳에 모입니다. 학생이 옆자리에서 점수를 올리면 내 화면의 순위표가 새로고침 없이 바뀝니다. 그리고 그 데이터베이스 앞에는 우리가 직접 세운 문지기가 서 있습니다.

이 단원에서 우리는 먼저 순위표를 만들지 않고 정하는 일부터 했습니다. 순위표

가 동기를 낮추기도 한다는 연구를 근거로 다섯 가지 설계 원칙을 세웠고, 그 원칙이 마지막에 프롬프트의 문장이 되어 화면으로 돌아왔습니다. 그다음 Firebase 콘솔을 열어 프로젝트를 만들고 애널리틱스를 껐으며, Firestore에서 컬렉션과 문서와 필드를 손으로 한 번 만들어 보았습니다. 무엇을 저장하고 무엇을 저장하지 않을지 네 칸으로 못 박았고, 기본으로 켜지는 Google 로그인을 익명 로그인으로 바꿔 개인정보를 아예 받지 않는 구조를 만들었습니다.

그리고 이 단원의 중심인 보안 규칙을 읽었습니다. "allow read, write: if true"가 무슨 뜻인지 알아보고, create만 열고 update와 delete는 막는 규칙으로 고쳐 게시했으며, 플레이그라운드에서 네 가지를 시험해 실제로 걸리는지 확인했습니다. 마지막으로 앱에 Firebase를 붙이고 TOP 10 리더보드와 교사용 회차 패널을 만들었으며, 사용량 화면에서 읽기가 먼저 닳는 것을 눈으로 보았습니다.

여기까지가 여섯 개 단원의 전부입니다. 1·2단원에서 게임을 보는 눈과 MDA를 익혔고, 3단원에서 AI Studio에 처음 접속해 첫 게임을 만들었으며, 4단원에서 성취기준에서 출발해 교과 맞춤 게임을 설계하고 플레이테스트했습니다. 5단원에서 그 게임에 주소를 달았고, 6단원에서 우리 반의 기록을 한곳에 모았습니다. 이제 선생님은 학생용 학습 게임을 처음부터 끝까지 혼자 만들어 배포할 수 있습니다.

남은 숙제. 만든 순위표를 다음 차시에 실제로 한 번 써 보고, 139쪽의 원칙 다섯 가지 중 무엇이 지켜졌고 무엇이 무너졌는지 한 문장씩 기록해 두세요. 그 기록이 다음 수업의 설계도가 됩니다.

더 나아가려면. 학급 간 협동 모드, 어느 문항에서 많이 틀렸는지 보여주는 교사 대시보드, 학생이 직접 문항을 만드는 편집 모드—모두 이 단원에서 익힌 것과 같은 방식으로 붙일 수 있습니다. 컬렉션 하나를 더 만들고, 규칙 몇 줄을 더 쓰면 됩니다.

이제 선생님이 만드는 차례입니다. 교실에서 실제로 한 번 돌려 보고, 그 기록을 다음 선생님에게 넘겨주세요.

참고문헌

Firebase. (2026). *Add Cloud Firestore and Authentication to your Google AI Studio app.* Firebase Docs. https://firebase.google.com/docs/ai-assistance/ai-studio-integration

Firebase. (2026). *Authenticate with Firebase anonymously on the web.* Firebase Docs. https://firebase.google.com/docs/auth/web/anonymous-auth

Firebase. (2026). *Get started with Cloud Firestore Security Rules.* Firebase Docs.

https://firebase.google.com/docs/firestore/security/get-started

Google. (2026). *Deploying from Google AI Studio.* Gemini API 개발자 문서. https://ai.google.dev/gemini-api/docs/aistudio-deploying

Hanus, M. D., & Fox, J. (2015). Assessing the effects of gamification in the classroom. *Computers & Education, 80,* 152–161.

Ames, C. (1992). Classrooms: Goals, structures, and student motivation. *Journal of Educational Psychology, 84*(3), 261–271.

교육부. (2024). *교육(지원)청·학교 개인정보 보호 가이드라인.*

개인정보보호위원회. (2023). *개인정보 보호법 제22조의2(만 14세 미만 아동).*

화면 캡처: Firebase Console(console.firebase.google.com) 및 Google AI Studio, 2026년 8월 25일·한국어 UI 기준. 구글 정책에 따라 화면 구성은 변경될 수 있습니다.

메모

메모

메모

코딩을 배운 적이 없어도
우리 반에 필요한 게임은 만들 수 있습니다.

전 국민의 게임 이용률이 50.2%인 데 비해 10대는 80.9%에 이릅니다. 학생에게 게임은 여가 중 하나가 아니라 생활 공간 그 자체입니다. 이 책은 그 공간을 금지의 대상으로 두는 대신, 교사가 직접 만드는 대상으로 바꾸자고 제안합니다.

AI 에이전트에게 한국어로 설명하면 작동하는 웹 게임이 만들어집니다. 교사가 할 일은 코드를 쓰는 것이 아니라 무엇을 만들지 정확히 말하는 것입니다. 그것은 원래 교사가 가장 잘하는 일입니다.

이 책을 덮을 때 남는 것

교과 내용을 담은 학습 웹 게임 한 편
학생이 링크와 QR로 바로 여는 배포본
우리 반의 기록이 실시간으로 보이는 순위표

여섯 단원

1 게임리터러시의 이해·2 AI 에이전트와 게임 개발·3 AI Studio 첫걸음
4 교과 맞춤 학습 게임·5 온라인 출판·6 실시간 리더보드

게임문화재단 | 게임문화교육원 Game Culture Education Institute | (사)한국창의정보문화학회

ISBN 979-11-24801-05-5
13370
9 791124 801055

비매품

2026 교사 게임리터러시 사업
게임문화재단·게임문화교육원·(사)한국창의정보문화학회