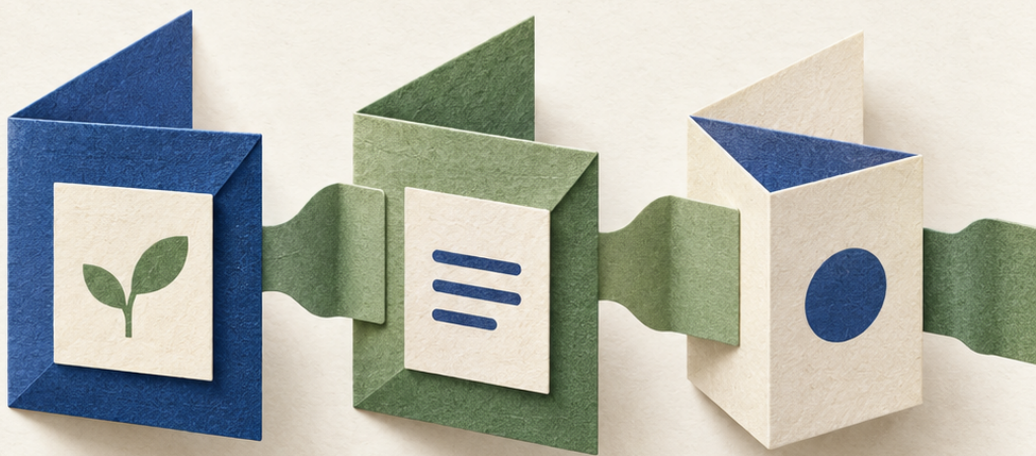


나만의 AI 스킬 만들기

반복 업무를 저장하고 실행하고 고치는
실무 가이드



Coreline-AI

3개의 스킬 · 1개의 종합 실습

INTRO / 읽기 안내

나만의 AI 스킬 만들기

반복 업무를 저장하고 실행하고 고치는 실무 가이드

매번 설명하는 일에서 다시 쓰는 일로

이 책은 AI를 사용하지만 같은 요청을 다시 설명하고, 결과를 고치고, 어디까지 말했는지 되짚는 사람을 위한 제작 실습서다. 제목 스킬, 작은 계획 스킬, 한 페이지 안내문 스킬을 만들고 마지막에 작은 업무 가이드를 끝까지 제작한다.

스킬을 많이 모으는 것보다 하나의 일을 정확히 작게 만드는 데 집중한다. 입력을 정하고, 결과 기준을 쓰고, 실패를 분류하고, 새 사본을 검토하는 과정이 책의 중심이다. 코드가 필요한 부분은 작은 파일 검사와 문서 변환으로 제한한다. 프로그래밍 언어나 애플리케이션 개발 전반을 가르치는 책이 아니다.

가상 사례로 판단을 설명하고 실제 합성 시험·로컬 도구 실행으로 확인한 범위를 구분한다. 그 구별 자체가 자신의 업무에 스킬을 적용할 때 필요한 기술이다.

Coreline-AI 지음

무료 PDF 제공용 · 2026년 10월

INTRO / 읽기 안내

이 판을 읽기 전에

저자 Coreline-AI · 나만의 AI 스킬 만들기 · 반복 업무를 저장하고 실행하고 고치는 실무 가이드. 무료 PDF 제공용으로 준비한 판이다. 실제 발행일은 최종 배포 확인 후 안내하며, 출판사·ISBN을 임의로 부여하지 않는다.

제품 사실은 가까운 공식 링크와 확인일로 범위를 표시했다. 정책과 화면은 달라질 수 있어 실제 사용 전 공식 안내를 다시 확인한다. 특정 요금이나 모든 환경의 호환을 약속하지 않는다.

인물·규칙·수치는 별도 실제 근거가 없으면 교육용 가상이다. 실제 합성 시험은 특정 환경의 관찰이며 실사용자 성과 통계가 아니다. 문서 변환 재현성과 새 AI 입력의 답변 검증도 다르다.

도해는 설명용이며 실제 제품 화면을 흉내 내지 않았다. 표지 후보는 내장 이미지 생성 도구로 제작했다. 제작 출처와 글꼴 고지를 기록했으며 제3자 자료는 해당 조건을 따른다. 무료 제공은 수정·재배포·재판매 허락이 아니다.

제품명은 식별을 위한 것으로 제휴·인증·공식 발행을 뜻하지 않는다. 법률·보안·조직 결정을 예제로 대신하지 않는다.

실습 묶음은 자동 설치 도구가 아니다. 민감 자료와 비밀값을 넣지 않고 허용한 사본으로 연습한다. 외부 발송·계정 설정·권한 확대·비용 발생·공개·판매는 별도 결정이다.

INTRO / 읽기 안내

목차 · 시작과 첫 제작

1부 · 반복 설명을 스킬로 바꾸기

- 01. 매번 같은 설명을 반복하고 있다면 — 업무 후보와 제외 이유를 고른다.
- 02. 스킬은 무엇을 묶고, 무엇을 하지 않는가 — 요청·규칙·스킬·도구의 역할을 나눈다.

2부 · 첫 번째 스킬 완성하기

- 03. 제목을 다듬는 첫 스킬 만들기 — 작은 SKILL.md와 근거 있는 후보를 만든다.
 - 04. 입력과 예시로 결과의 기준을 세우기 — 입력·예시·기대 조건을 연결한다.
 - 05. 같은 스킬을 다시 써보고 고치기 — 실패·개정·재시험과 의미 한계를 기록한다.
- PDF 북마크와 EPUB 탐색 목차에서 각 장을 바로 찾을 수 있다. EPUB 쪽 수는 리더 설정에 따라 달라진다.

INTRO / 읽기 안내

목차 · 업무 적용과 복구

3부 · 실제 업무에 쓰는 스킬 만들기

06. 일을 끝낼 범위와 순서 정하기 — 목표·제외·선행조건·완료를 나눈다.

07. 참고 자료와 양식을 필요한 만큼 나누기 — 언제 읽고 언제 멈출지 연결한다.

08. 메모를 한 페이지 안내문으로 바꾸기 — 사실·판단·HTML 표현을 나눈다.

09. 필요한 곳에만 실행 도구 붙이기 — 결정적인 검사만 작은 도구에 맡긴다.

4부 · 실패하는 스킬 고치기

10. 결과가 틀렸을 때 고칠 곳 찾기 — 입력·지시·자료·도구의 원인을 분류한다.

11. 안전하게 실행하고 다른 환경으로 옮기기 — 경로·권한·비용·환경을 확인한다.

INTRO / 읽기 안내

목차 · 결과를 완주하기

5부 · 여러 스킬로 결과물 완성하기

12. 여러 스킬을 하나의 일로 연결하기 — 일곱 역할의 파일과 상태를 인계한다.

13. 작은 업무 가이드를 끝까지 제작하기 — 실제 여덟 쪽 변환과 실패·재개를 시험한다.

14. 검수하고 다시 만들 수 있게 남기기 — ZIP 재열기·새 폴더 재빌드·사람 결정을 나눈다.

뒤에서 다시 찾을 안내

작은 결과를 다음 반복으로 넘기기 / 내가 쓰는 말로 바꾸기: 파일·상태 / 환경이 다르면 먼저 볼 것 / 작은 실패 뒤 돌아가는 길 / 공식 자료와 교육용 예제 / 실습 묶음의 파일 안내 / 새 입력으로 다시 시험하기

본문 설명을 읽은 뒤 필요한 말과 실행 조건, 출처와 연습 위치를 다시 찾는 안내다.

INTRO / 읽기 안내

읽는 순서를 고르는 법

처음 만드는 독자라면

01~05장을 순서대로 읽으며 자신의 반복 업무 하나를 제목 연습에 대입해 보자. 첫날에는 SKILL.md의 많은 기능을 외우지 않아도 된다. 언제 사용할지, 어떤 입력을 읽을지, 무엇을 남기고 무엇을 하지 않을지 자신의 말로 쓸 수 있으면 시작할 수 있다.

책의 완성 예시를 바로 복사하기 전에 빈 양식을 먼저 읽는다. 빈칸은 모르는 값을 상상해 채우는 자리가 아니다. 실제로 아는 조건과 아직 물어야 할 조건을 나누는 자리다. 작은 시작본은 원리를 배우는 파일이며, 기존 실제 시험을 통과한 선택 개정본과 같은 버전이 아니다.

이미 스킬을 쓰는 독자라면

반복 실패가 무엇인지 적고 10장의 분류로 원인을 나눠 보자. 결과를 고치려 지시를 늘렸는데 실제로는 자료가 빠졌을 수도 있다. 11장에서는 파일이 읽히지 않는 문제와 권한을 구별한다. 제품별 설치 위치를 기억하기 보다 현재 공식 안내와 자신의 허용 범위를 먼저 확인한다.

06~09장은 계획·참고 자료·HTML·검사 도구의 경계를 구체적으로 보여 준다. 기존 지시가 길다면 07장의 파일 분리 기준을 적용하고, 도구를 무조건 붙이고 있다면 09장의 작은 검사를 먼저 따라 해 보자. 어느 경우든 실제 입력 하나를 끝까지 읽으며 행동과 검토를 연결한다.

마지막 결과가 먼저 궁금하다면

13장의 여덟 쪽 가이드를 펼쳐 보고 12장의 역할 분담으로 돌아갈 수 있다. 다만 원고 변환 성공을 스킬의 새 업무 판단 검증으로 읽지 않는다. 어떤 단계가 실제 실행이고 어떤 사례가 가상인지 확인한 뒤 14장의 새 폴더 재빌드를 수행한다.

책은 한 가지 질문을 반복한다. ‘이 결과로 사람이 다음에 무엇을 할 수 있는가?’ 결과가 멋져 보여도 자료가 없거나 승인 조건이 모호하면 다음 행동을 확정하지 않는다. 각 장의 완성 카드와 빈 양식은 그 질문에 답하기 위한 보조물이며 설명을 대신하는 정답 목록이 아니다.

INTRO / 읽기 안내

실습을 시작하는 작은 준비

원본 대신 연습 사본을 만든다

새 로컬 폴더를 하나 만든다. 고객 자료나 회사 계정을 연결하지 않고 책의 비민감 입력과 자신의 가상 메모로 시작한다. 받은 원본은 보존하고 작업 사본을 따로 둔다. 설치·발송·업로드·Git 변경을 실습 목적에 섞지 않는다. 파일 읽기와 명령 실행도 범위를 확인한 뒤 진행한다.

교육용 E01은 제목, E02는 작은 계획, E03은 한 페이지 안내문, E04는 작은 가이드다. 원본 프로젝트 전체를 독자 폴더에 복제한 것이 아니라 참고 원리를 새 예제로 구성했다. 스킬 파일이 들어 있다고 자동 활성화되는 묶음이 아니므로 실행 전에는 사용 제품의 현재 검색 위치와 호출 방식을 확인한다.

도구를 한 번에 설치하지 않는다

스킬의 지시를 읽고 수정하는 단계와 Python 검사기를 실행하는 단계를 나누자. 텍스트 편집만으로 이해할 부분이 있다. HTML 기본 검사는 Python 표준 라이브러리로 수행한다. E04 문서 변환에는 ReportLab과 pypdf가 필요하다. 필요한 패키지가 없다면 별도 환경 준비를 확인한다. 책의 명령은 자동 설치나 비용 승인을 포함하지 않는다.

실제 제작 검증 환경은 macOS, Python 3.12.14, ReportLab 4.4.9, Codex CLI 0.145.0이다. 제품 사실은 2026-10-03 공식 자료를 확인했

다. 다른 운영체제와 모델의 동일 행동을 보장하지 않는다. EPUB 실제 좁은 화면·큰 글자 검수는 이 로컬 판에서 미수행으로 남겨 둔다.

실행 결과는 짧게라도 남긴다

입력, 선택 스킬 버전, 기대 조건, 실제 출력, 판단, 다음 행동을 한 카드에 적는다. 프로그램의 종료 값 0은 지정한 검사를 끝냈다는 뜻이며 업무 사실 전체를 승인한 값이 아니다. 새 AI 호출을 하지 않았다면 파일 검사를 새 행동 시험이라고 기록하지 않는다.

E01의 근거 없는 성공 수식어와 E02의 자료 부족·완성 상태 문제는 강화 의미 기준의 새 실행 검토가 남는다. 실습 묶음에서도 선택 역사 버전과 수정 제안을 구별한다. 이 한계를 읽고 무리 없는 연습 범위로 사용하자. 도구의 한계를 아는 것이 실패를 숨기는 것보다 더 좋은 출발이다.

PART 1 / 다음 제작으로

반복 설명을 스킬로 바꾸기

반복을 발견하는 것과 위임할 수 있는 일을 고르는 것은 다르다. 첫 부에서는 같은 사실과 같은 작업 방식을 나누고, 스킬·요청·규칙·도구의 역할을 구별한다. 일의 이름을 고르기 전에 결과와 제외 이유를 적는다.

남길 것은 후보 카드와 최소 역할 설명이다. 아직 설치나 복잡한 파일 구조로 들어가지 않는다. 반복되는 한 업무를 떠올리고 어떤 부분에서 매번 설명이 길어지는지 읽어 보자.

01 / 반복 설명을 스킬로 바꾸기

매번 같은 설명을 반복하고 있다면

어떤 일을 스킬로 만들 가치가 있는가?

이 장에서 남길 것: 업무 후보와 제외 이유를 적은 설계 카드

지난주에 만든 안내문을 열었는데 어디를 고쳐야 할지 바로 보이지 않는다. 날짜는 바꿨고 장소도 바꿨다. 그런데 준비물에 지난 행사의 내용이 남아 있다. 처음부터 새로 쓰면 이런 실수는 줄 것 같지만, 이번에는 대상 독자와 말투, 확인되지 않은 정보의 처리 방법을 AI에게 다시 설명해야 한다. 예전 대화에서 잘 나온 답을 찾아 붙여 넣는 동안, 어느 조건이 지금도 유효한지 더 헛갈린다.

이런 일을 겪었다고 해서 곧바로 큰 자동화가 필요한 것은 아니다. 먼저 구별해야 할 것은 **이번에 바뀐 자료와 매번 지키고 싶은 작업 방식**이다. 새 날짜와 준비물은 이번 자료다. 제공된 정보만 쓰고, 모르는 것은 물어 보고, 짧은 초안을 만든 뒤 사람이 확인한다는 약속은 작업 방식이다. 둘을 매번 한 덩어리로 다시 말하면, 사실도 규칙도 놓치기 쉽다.

이 책에서 만드는 AI 스킬은 반복하는 작업 방식을 파일에 남겨 필요한 때 다시 쓰는 데서 출발한다. 모든 업무를 스킬로 바꾸지는 않는다. 처음에는 작고, 틀려도 원본을 해치지 않으며, 결과를 사람이 살펴볼 수 있는 일을 고른다. 이 장에서는 파일을 설치하거나 프로그램을 실행하지 않는다. **만들 가치가 있는 업무 하나와, 아직 맡기지 않을 일을 고르는 것**이 목표다.

반복하는 것은 문장인가, 판단인가

매주 쓰는 공지가 모두 같은 문장일 필요는 없다. 대상과 일정이 달라지면 결과 문장은 달라지는 편이 자연스럽다. 다시 쓰고 싶은 것은 “누가 읽는지 확인한다”, “일정이 서로 다르면 묻는다”, “초안과 발송을 구분한다” 같은 판단이다. 결과를 고정하는 것과 작업 방식을 고정하는 것은 다르다.

이를 구별하지 않으면 지난 공지를 복사하는 것과 스킬을 쓰는 것이 비슷해 보인다. 복사한 공지는 지난번의 사실까지 함께 가져온다. 작업 방식만 남긴 스킬은 새 사실을 받아 같은 순서로 살펴보도록 만든다. 하지만 파일에 그렇게 적었다고 해서 AI가 언제나 정확히 수행하는 것은 아니다. 실제 입력을 보내 보고, 무엇을 빠뜨렸는지 확인하는 일이 뒤따라야 한다. 이 책이 제작뿐 아니라 시험과 수정까지 다루는 이유다.

예를 들어 지난 안내에 “금요일 오후, A회의실”이라고 쓰여 있었다고 하자. 이 말을 다음 주에도 유지할지 판단할 자료가 없다면, 그대로 가져오면 안 된다. 재사용할 수 있는 규칙은 “일정과 장소는 이번에 제공된 자료에서 확인한다”이지 “금요일 A회의실을 기본값으로 삼는다”가 아니다. 작은 차이지만 첫 번째는 사실을 확인하는 습관을 남기고, 두 번째는 오래된 사실을 반복한다.

다시 쓸 것과 새로 받을 것

통째로 다시 쓰기

지난 공지: 사실 + 말투 + 작업 방식
이번 자료로 고쳤는지 다시 확인

나눠서 다시 쓰기

이번 자료: 대상 · 일정 · 장소 · 준비물
작업 방식: 확인 → 초안 → 사람 검토

새 사실을 넣어 새 초안을 만든다.
작업 방식을 저장해도 결과 검토는 남는다.

그림 01-1. 무엇을 다시 쓰고 무엇을 새로 받아야 하는가? 날짜·장소 같은 사실은 새로 받고, 확인·작성·검토의 순서만 재사용한다. 저자의 설명용 도형이며 제품 화면이 아니다.

반대로 반복할 판단이 거의 없는 일도 있다. 오직 한 번만 필요한 짧은 문장이고, 결과 조건도 단순하다면 그때의 요청만으로 충분할 수 있다. 스킬 파일을 만드는 일에는 읽고 고치는 수고가 생긴다. 반복되지 않는 일을 위해 그 수고를 먼저 만들 필요는 없다. ‘스킬이 있으면 더 전문적’이라는 생각보다 ‘다음번에도 같은 판단을 꺼내 쓸까?’라는 질문이 낫다.

이 장을 끝내면 남을 한 장

첫 작업물은 스킬 자체가 아니라 **업무 후보 카드**다. 카드에는 어떤 입력을 받아 무엇을 만들지, 결과를 누가 확인할지, 무엇을 하지 않을지를 적는다. 이 카드가 분명해야 다음 장에서 파일의 역할을 이해하고, 03장에서 첫 스킬을 만들 때 일을 필요 이상으로 늘리지 않을 수 있다.

카드의 핵심은 거창한 목표가 아니다. “팀 소통을 자동화한다”는 목표는 공지 작성, 일정 조정, 메일 발송, 반응 수집까지 끝없이 넓어질 수 있다. 반면 “이번 주 교육 메모에서 공지 초안 한 개를 만든다”는 말에는 입력과 결과가 있다. 사람이 검토한 초안을 발송하는 일은 따로 남겨 둘 수 있다. 작은 목표는 대단하지 않아서 좋은 것이 아니라, **잘했는지 확인할 수 있어서** 좋다.

작성할 때에는 ‘매주 쓰니까 필요하다’에서 멈추지 않는다. 어떤 설명을 되풀이하는지 한 문장 더 적는다. 새 직원에게 쉬운 말로 설명하라는 조건인지, 날짜가 없으면 확정하지 말라는 조건인지, 사내 문서를 외부 주소로 바꾸지 말라는 조건인지 살펴본다. 실제 반복 조건을 쓰지 못하면 업무 이름만 적힌 카드가 된다. 그런 카드로 스킬을 만들면 무엇을 재사용해야 하는지 다시 추측해야 한다.

뒤쪽의 가상 팀 카드와 빈 양식에 자기 업무를 대입해 보자. 기술 용어보다 후보를 먼저 고르는 것은 **잘못 고른 일을 정확하게 구현하는 낭비를 피하기** 위해서다.

바뀌는 자료와 남길 약속을 갈라 놓기

책상 위의 업무 하나를 떠올려 보자. 같은 종류의 결과를 지난번에도 만들었다면, 지난 요청과 이번 요청을 나란히 놓는다. 이번에만 맞는 내용에는 ‘이번 자료’, 다음에도 지켜야 할 처리 방법에는 ‘작업 약속’이라고 표시해 본다. 이것은 이 책의 설계 연습이지 제품이 제공하는 분류 기능은 아니다.

“토요일 오전에 진행한다”는 이번 자료다. “날짜가 없으면 임의로 넣지 않는다”는 작업 약속이다. “처음 참석하는 사람에게 보낸다”는 이번 자료일 수도 있고 계속되는 독자 조건일 수도 있다. 상황에 따라 다르면 입력

칸에 두는 편이 낫다. 처음부터 어느 칸이 정답인지 맞추려 하지 말고, 다른 독자가 들어왔을 때 결과가 어떻게 달라져야 할지 생각해 보자.

남겨 둘 약속은 관찰할 수 있는 행동으로 적는다. “정확하고 친절하게”는 방향을 알려 주지만 무엇을 확인할지 충분히 말해 주지 않는다. “제공된 장소만 쓰고, 장소가 없으면 확인 필요라고 표시한다”는 행동과 예외를 함께 말한다. “글을 잘 써라”보다 “독자가 해야 할 행동을 초안 앞부분에서 찾을 수 있게 한다”가 결과를 검토하기 쉽다. 구체적인 입력과 예시를 더하는 방법은 04장에서 다룬다.

모든 조건을 고정하려는 것도 문제다. 안내를 반드시 다섯 문장으로 쓰게 하면 짧은 내용에는 맞아도 일정 변경이나 준비물 설명이 많은 날에는 중요한 정보가 빠질 수 있다. 처음에는 결과의 모양보다 내용의 경계를 고정한다. 제공된 사실을 빠뜨리지 않기, 없는 사실을 만들지 않기, 초안으로만 남기기 같은 약속이 먼저다. 문장 수는 그 일을 시험해 본 뒤 조정해도 된다.

여기서 한 가지를 더 나눈다. 자료가 없을 때 던질 질문과, 자료가 있는데 서로 다른 때 던질 질문은 다르다. 장소가 없다면 “어디에서 진행하나요?”라고 물으면 된다. 두 문서의 장소가 다르다면 어떤 문서가 이번 행사에 적용되는지 확인해야 한다. 최신처럼 보이는 파일 이름만으로 하나를 고르면, 오래된 공지가 새 자료를 덮어쓸 수 있다. 후보 카드에는 적어도 이런 예외 한 가지를 남기는 것이 좋다.

좋은 첫 후보에는 돌아갈 길이 있다

첫 스킬을 고를 때에는 결과가 틀렸을 때 무엇을 잃는지 생각한다. 로컬 초안 한 개라면 틀린 부분을 고치거나 버리고 다시 만들 수 있다. 원본 메모를 보존하면 어디서 틀렸는지도 대조할 수 있다. 반면 공지를 바로 발송하거나 파일을 삭제하는 일은 초안과 같은 방식으로 취급할 수 없다. 사람과 시스템에 이미 영향을 주었기 때문이다.

그래서 첫 후보는 **입력을 볼 수 있고, 결과를 비교할 수 있고, 틀리면 멈출 수 있는** 일이 좋다. 이 세 표현은 공식 인증 기준이나 점수표가 아니다. 입문자가 시행착오를 감당할 수 있는 범위를 고르기 위한 저자의 질문이다. 어떤 일은 입력이 잘 정리돼도 결과를 확인할 사람이 없고, 어떤 일은 반복 빈도가 높아도 결과의 책임이 너무 크다. 하나의 숫자로 더해서 선정하지 않는다.

입력을 볼 수 있다는 것은 파일 이름을 안다는 뜻만이 아니다. 필요한 내용에 접근할 수 있어야 하고, 그 자료를 해당 도구에 입력해도 되는지 담당자가 판단해야 한다. 이 책의 첫 실습에는 공개해도 되는 합성 메모를 쓴다. 회사 자료를 반드시 가져오라는 요구는 없다. 원본 문서를 열 권한이 없는데 스킬이 내용을 읽었다고 보고한다면, 성공한 업무가 아니라 확인되지 않은 주장이다.

결과를 비교할 수 있다는 것은 ‘마음에 든다’보다 조금 더 구체적이다. 제목 후보라면 본문에 없는 성과가 들어갔는지 볼 수 있다. 안내 초안이라면 새 날짜와 장소가 원본 메모와 같은지 대조할 수 있다. 계획이라면 담당자와 순서가 실제 조건에 맞는지 검토할 수 있다. 반대로 ‘조직 전체의 업무 품질 개선’처럼 결과를 확인할 방법이 흐릿하면 작은 산출물로 다시 잘라야 한다.

틀리면 멈출 수 있다는 것은 실패가 발생하지 않는다는 약속이 아니다. 필요한 자료가 없으면 질문만 남기고, 서로 다른 사실을 만났으면 초안을 확정하지 않으며, 사람의 확인 전에는 외부로 보내지 않는다는 뜻이다. 잘못된 답을 만들 가능성을 완전히 없앨 수는 없어도, 잘못된 답이 곧바로 행동으로 이어지는 길은 줄일 수 있다.

첫 후보를 고르는 세 가지 질문

1. 필요한 자료를 볼 수 있는가?

자료 없음·접근 불가면 먼저 확인

2. 결과를 사람이 비교할 수 있는가?

원본 사실과 초안의 대응을 대조

3. 모르는 상황에서 멈출 수 있는가?

질문·확인 필요 표시 후 외부 행동 없음

막힌 곳이 있으면 범위를 줄여 다시 고른다.

저자의 입문 설계 질문 · 공식 점수표 아님

그림 01-2. 이 일을 첫 스킬로 삼아도 될까? 필요한 자료를 볼 수 있는지, 결과를 사람이 대조할 수 있는지, 모르는 상황에서 멈출 수 있는지를 순서대로 묻는다. 저자의 입문 설계 기준이며 공식 평가 점수가 아니다.

자주 하는 일이라도 처음에는 제외한다

반복 빈도가 높다고 모두 적합한 것은 아니다. 가상의 팀에서 직원 평가를 매주 정리한다고 해 보자. 문장을 요약하는 도움과 사람의 평가를 자동화하는 일은 다르다. 최종 평가를 정하는 책임을 초보자의 첫 스킬에 넘기는 것은 이 책의 실습 범위가 아니다. 반복성을 보았다고 판단의 권한까지 맡기지는 않는다.

‘고객 문의 답변’도 한 덩어리로 고르면 너무 넓다. 공개된 FAQ에서 답변 초안을 정리하는 것, 내부 주문 정보를 읽는 것, 고객에게 보내는 것은 서로 다른 행동이다. 첫 연습에서는 가상 문의와 공개해도 되는 예시 답변으로 초안까지만 만들 수 있다. 실제 고객 정보 접근이나 전송까지 포함하려면 자료·권한·검토 조건을 별도로 다뤄야 한다. 여기에서는 그런 행동을 승인했다고 가정하지 않는다.

처음부터 포기할 필요는 없다. 넓거나 위험한 일을 발견했다면 그 안에서 작은 준비 작업을 분리한다. 고객에게 보내기 대신 답변 초안 만들기, 일정 확정 대신 충돌 목록 정리하기, 원본 수정 대신 변경 제안 쓰기가 가능하다. 다만 분리한 작업도 실제로 검토할 수 있어야 한다. 이름에 ‘초안’을 붙였지만 자동 발송 기능을 함께 두면 범위는 여전히 넓다.

가상 팀은 공지 작성에서 어디를 잘랐을까

이 절의 팀, 업무 메모, 장소와 결정은 모두 **교육용 가상 사례**다. 실제 조직을 조사하거나 새 스킬을 실행한 결과가 아니다. 팀의 교육 담당자는 매주 새 참석자에게 시설 이용 안내를 만든다. 같은 설명을 반복하지만, 일정과 준비물은 매번 바뀐다. 담당자는 우선 ‘공지 작성부터 발송까지’를 후보로 적었다가 너무 넓다는 것을 알게 된다.

이번에 제공된 메모는 이렇다. 읽을 사람은 처음 참석하는 팀원이다. 금요일 오후에 안내를 진행하고, 이번 주에는 B회의실을 쓴다. 준비물은 노트북이다. 이전 공지에는 A회의실이라고 적혀 있다. 담당자는 아직 ‘이전 공지’를 말투 참고 자료로 쓸지 사실 자료로 쓸지 구분하지 않았다. 두 문서를 그대로 보내면 AI는 어느 장소를 따라야 할까?

후보를 좁힌 뒤 처리 순서를 정했다. 이번 행사에 대한 메모를 사실 자료로 삼고, 이전 공지는 표현 참고로만 읽는다. 두 문서의 역할이 표시되지

않으면 장소를 추정해 채택하지 않는다. 사람이 확인해야 할 질문을 먼저 적는다. “이번 안내의 장소는 B회의실이 맞고, 이전 공지는 말투만 참고 하면 되나요?”라는 질문이면 필요한 결정을 한 번에 확인할 수 있다.

가상의 담당자는 B회의실이 맞다고 답하고, 이전 공지에서 일정과 장소를 가져오지 말라고 확인한다. 그다음에야 초안을 작성한다. 초안에는 금요일 오후, B회의실, 노트북 준비를 넣고, 제공되지 않은 시작 시각은 ‘정확한 시각 확인 필요’로 남긴다. 금요일 오후라는 말만 보고 2시라고 채우지 않는다. 자주 쓰던 시간이 있더라도 이번 자료에 없는 사실이기 때문이다.

담당자는 초안을 검토하면서 두 가지를 구별한다. 장소가 B회의실인 것은 원본과의 대조로 확인할 수 있다. 처음 참석하는 사람이 안내를 이해하는지는 문장을 읽어 판단해야 한다. 준비물은 있지만 어디로 들어와야 하는지 안내가 빠졌다면, 원본에서 추가 정보를 받아 수정한다. 스킬이 자료에 없는 건물 출입 동선까지 만들어 내도록 요구하지 않는다.

이 사례의 결과는 ‘공지 자동화 성공’이 아니다. **일정 충돌을 확인한 뒤 사람이 검토할 초안의 범위가 정해졌다는 것이다.** 발송은 후보에서 뺀고, 장소가 모호할 때는 질문으로 멈추며, 초안에 없는 사실을 확인하는 담당자가 남았다. 실제 효과는 나중에 스킬을 만들고 사용해 봐야 알 수 있다. 이 단계에서 시간을 얼마나 줄였다고 적을 근거는 없다.

후보를 카드로 남기면 모호한 부분이 보인다

가상 팀의 후보 카드를 완성해 보자. 아래 내용은 실제 실행 이력이 아니라 앞 사례에서 정한 설계 결과다.

항목	작성한 내용
업무 후보	이번 주 시설 이용 안내의 공지 초안 만들기
반복해서 설명하는 것	처음 참석하는 사람에게 쉽게 설명, 제공 사실만 사용, 미확정 정보 표시
매번 새로 받을 자료	대상, 일정, 장소, 준비물, 사실 자료와 표현 참고 자료의 구분
이번에 만들 결과	담당자가 검토할 공지 초안 한 개와 확인 질문
중단하고 물을 때	장소 충돌, 시작 시각 누락, 자료에 접근할 수 없음
사람이 결정할 것	이번 자료의 적용 여부, 사실 확인, 표현과 발송 여부
제외할 행동	이전 공지 원본 덮어쓰기, 참가자 정보 수집, 자동 발송
처음 시험할 입력	가상 정상 메모 한 건, 장소가 다른 메모 두 건
성공을 확인할 방법	초안을 메모와 대조하고 모르는 정보의 처리와 제외 행동을 확인
다음 재검토 계기	안내 방식이나 검토 담당자가 바뀔 때

‘이번에 만들 결과’를 한 개로 좁힌 것이 중요하다. 공지와 참가자 명단과 참석률 보고서를 동시에 만들기 시작하면, 어느 작업에서 잘못됐는지 찾기 어려워진다. 첫 카드에서는 초안과 그 초안에 필요한 확인 질문만 남긴다. 질문이 결과에 함께 있는 것은 실패가 아니라 자료를 확정하지 못했음을 보이는 방식이다.

‘성공을 확인할 방법’도 따로 적는다. 공지의 글자 수만 재면 핵심 일정이 빠진 초안도 통과할 수 있다. 원본과 대조해야 할 사실, 모르는 정보의 표시, 외부 행동을 하지 않았는지를 본다. 이 확인 방법은 이번 후보에 맞춘 것이며 모든 종류의 스킬에 같은 표를 강제하지 않는다. 제목, 계획, 안내 문은 확인할 결과가 서로 다르다.

카드를 다 쓰고도 입력 칸이 비어 있다면, 스킬 본문을 먼저 늘리는 대신 필요한 자료를 얻는 방법부터 정한다. 담당자가 어떤 문서를 주어야 하는지조차 모르면 파일 한 장의 지시로 그 문제를 없앨 수 없다. 접근할 자료가 없는데 완성된 작업물부터 약속하는 것이 대표적인 나쁜 시작이다.

잘 만든 파일도 돌볼 사람이 필요하다

처음 만드는 사람과 계속 사용하는 사람이 다를 수 있다. 파일을 공유받은 동료가 ‘왜 여기서 질문만 하고 멈추지?’라고 느낀다면, 스킬의 이름보다 제외 이유를 함께 남기는 편이 도움이 된다. 가상 팀의 카드에는 장소 충돌이 있었기 때문에 확인 질문이 필요하다는 맥락이 있다. 이유가 있어야 사용자가 규칙을 무심코 지우지 않는다.

담당자가 바뀌면 검토 지점도 확인해야 한다. 이전 담당자가 사실 자료를 주던 사람이었다면, 새 담당자는 문장만 고치는 역할일 수 있다. 그런 사람에게 일정 확정까지 맡기면 검토라는 이름만 남는다. 스킬을 만들기 전에 누가 어떤 정보에 대해 답할 수 있는지 적는 이유다. 이것은 기술 설정이 아니라 업무 역할의 확인이다.

시설 위치처럼 자주 바뀌는 사실은 입력으로 받더라도 운영 방식 자체가 바뀌면 카드와 스킬을 다시 읽어야 한다. ‘파일로 저장했으니 끝’이 아니라 다시 쓰고 다듬는 작업이다.

여러 번 불러 썼다는 사실도 정확성이나 시간 절약의 증거는 아니다. 무엇을 고쳤고 어디서 질문이 필요했는지 남긴다. 반복 시험과 개정 기록은 05장에서 구체화한다.

내 업무의 후보를 한 개만 고르기

자기 업무를 적을 때에도 처음부터 실제 자료를 가져올 필요는 없다. 공개해도 되는 가상 메모로 작업의 모양을 먼저 만든다. 특정 고객 이름이나 사내 비공개 문서를 복사하지 않아도 반복 조건은 찾을 수 있다. 중요한 것은 입력이 들어왔을 때 어떤 판단을 거쳐 어느 결과에서 멈출지 설명하는 것이다.

다음 양식을 채운 뒤 마지막 두 문장을 특히 읽어 보자. “자료가 없으면 무엇을 할까?”와 “하지 않을 행동은 무엇일까?”에 답할 수 없다면 아직 후보를 줄일 여지가 있다. 일이 자동으로 이어지도록 만들기 전에, 끊어야 할 곳부터 표시하는 연습이다.

업무 후보 이름:
반복해서 다시 설명하는 조건:
매번 새로 받을 자료:
이번에 만들 결과 한 가지:
결과를 검토할 사람과 확인 범위:
자료가 없거나 서로 다를 때 할 일:
외부 행동 없이 끝나는 지점:
하지 않을 행동과 제외 이유:
먼저 시험할 가상 입력:
이 후보를 다시 검토할 계기:

카드를 채웠는데 업무 이름이 여전히 ‘팀 운영 전체’라면 더 작게 나눈다. 결과가 ‘완벽한 안내’라면 담당자가 볼 수 있는 초안으로 바꾼다. 결과를 확인할 사람이 없다면 먼저 그 역할을 정한다. 이 수정은 계획을 약하게 만드는 일이 아니다. 무엇을 실제로 만들고 살펴볼 수 있는지 정하는 일이다.

다음 장에서는 한 번의 요청, 프로젝트 규칙, 스킬, 실행 도구가 어떻게 다른지 알아본다. 03장에서는 작은 제목 스킬을 만든다. 앞으로 새로운 기능을 만나도 **내 업무의 어느 부분을 맡는가부터 물어보자.**

02 / 반복 설명을 스킬로 바꾸기

스킬은 무엇을 묶고, 무엇을 하지 않는가

프롬프트·규칙·도구와 무엇이 다른가?

이 장에서 남길 것: 최소 폴더와 역할 설명

업무 후보를 골랐으니 긴 요청문을 파일로 저장하면 될 것 같다. 그런데 저장한 뒤에도 AI가 예전처럼 답한다. 파일 이름을 바꿔야 하는 걸까, 문장을 더 많이 적어야 하는 걸까? 반대로 어떤 사람은 지시서에 “안내문 파일을 확인한다”라고 썼으니 검사 프로그램까지 자동으로 생겼다고 생각한다. 두 경우 모두 **적어 둔 지시와 실제로 일어나는 행동**을 같은 것으로 본 데서 혼란이 시작된다.

앞 장에서 고른 것은 어떤 일을 맡길지에 대한 후보였다. 이번 장에서는 그 일을 어디에 적고 무엇으로 실행할지 구별한다. 한 번의 요청, 프로젝트에서 계속 지킬 규칙, 필요한 때 꺼내 읽는 스킬, 파일을 읽거나 검사하는 도구는 역할이 다르다. 이 차이를 알면 처음부터 모든 것을 하나의 파일에 밀어 넣거나, 코드가 없어도 되는 일에 프로그램을 붙이지 않을 수 있다.

이 장의 작업물은 **최소 스킬 폴더와 그 안팎의 역할 설명**이다. 완성된 제목 스킬의 문장은 03장에서 직접 쓴다. 지금은 파일 구조를 읽을 수 있고, “이 파일은 규칙을 설명하지만 이 행동은 아직 실행하지 않았다”라고 말할 수 있으면 된다. 새 도구를 많이 갖추는 장이 아니라, 서로 다른 것을 같은 이름으로 부르지 않는 장이다.

같은 업무에도 네 가지 역할이 있다

가상의 담당자가 팀 행사 안내를 준비한다고 해 보자. “이번 행사는 새 팀원을 위한 것이고, 금요일 오후에 진행한다”라는 말은 이번 작업의 입력이다. “제공하지 않은 일정은 추정하지 않는다”는 말은 여러 작업에 계속 적용하고 싶은 약속이다. “안내 요청이면 자료를 확인하고 초안을 작성한 뒤 누락 정보를 보여 준다”는 말은 특정 업무의 순서다. 마지막으로 HTML 파일이 있는지 실제로 확인하는 프로그램은 그 순서의 일부를 수행하는 도구다.

우리는 이 역할을 각각 **이번 요청**, **프로젝트 공통 규칙**, **업무 스킬**, **실행 도구**라고 나누어 생각한다. 이 구분은 책을 읽기 위한 설명 모델이지 모든 제품이 똑같은 이름과 우선순위로 구현한다는 뜻이 아니다. 중요한 것은 어느 정보가 이번에 바뀌고, 어느 절차가 재사용되며, 무엇이 실제 행동을 일으키는지다.

역할	담는 내용	잘못 이해하기 쉬운 점
이번 요청	이번 자료, 원하는 결과, 이번엔 허용한 행동	지난번 자료도 자동으로 맞다고 생각함
프로젝트 공통 규칙	여러 작업에서 계속 지킬 기본 약속	모든 업무의 상세 순서를 한꺼번에 넣음
업무 스킬	특정 요청에 맞는 처리 순서와 예외	저장만 하면 모든 대화에 적용된다고 생각함
실행 도구	파일 읽기·쓰기·검사 등 실제 작업	실행 성공을 사실·품질 승인으로 생각함

예를 들어 안내문과 회의록 모두 원본을 보존해야 한다면, 원본 보존은 공통 약속으로 생각할 수 있다. 안내문에서 일정과 준비물을 확인하는 순서는 해당 업무에 가깝다. ‘금요일’이라는 사실은 이번 요청에 들어간다. 파일 존재를 검사하는 것은 실행 도구에 맡길 수 있지만, 안내문의 날짜가 맞는지 원본 자료와 사람이 판단해야 한다.

이렇게 나누면 비슷한 문장을 여러 곳에 반복해서 쓰는 일도 줄일 수 있다. 공통 규칙에 상세 안내문 절차를 길게 적고, 같은 절차를 스킬에도 적고, 매번 요청에도 붙이면 어느 부분을 고쳐야 하는지 모호해진다. 한 곳의 문장만 고쳐도 다른 곳에는 옛 내용이 남는다. 처음부터 완벽한 분리를 할 필요는 없지만, 같은 약속의 기준이 어디에 있는지는 설명할 수 있어야 한다.

스킬은 필요한 때 읽는 업무 묶음이다

현재 OpenAI의 공식 안내는 스킬을 [SKILL.md](#) 중심의 폴더로 설명한다. 지시문과 이름·설명이 들어가는 파일이 필수이고, 스크립트·참고 자료·양식 같은 자원은 필요한 경우에 더할 수 있다. 처음부터 코드나 여러 보조 파일이 있어야 스킬이 되는 것은 아니다. [OpenAI 공식 Build skills, 2026-10-03 확인](#).

쉽게 말하면 [SKILL.md](#)는 업무 카드이고 폴더는 카드와 관련 자료를 담은 서랍이다. 카드에는 이 서랍이 어떤 요청에 맞는지, 꺼낸 뒤 어떤 순서로 처리할지 적는다. 서랍 안의 참고 자료는 판단에 필요한 정보이고, 양식은 만들 결과의 기본 모양이다. 검사 코드가 있다면 특정 확인을 실제로 수행한다. 서랍만 있다고 자료가 읽히거나 프로그램이 실행되는 것은 아니다.

스킬의 이름과 설명을 먼저 보고, 선택한 뒤 본문을 읽는 방식도 공식 안내에 있다. 필요한 정보를 단계적으로 읽는 것을 ‘점진적 공개’라고 부르지만 용어보다 이유를 기억하면 된다. 모든 업무 지시서를 처음부터 한꺼번에 읽기보다, **지금 요청에 맞는 지시서부터 자세히 읽는 것이다**. 이름과 설명이 모호하면 첫 선택부터 틀릴 수 있다. [OpenAI 공식 Build skills의 사용 방식, 2026-10-03 확인](#).

따라서 “뭐든 잘 해주는 스킬”이라는 설명보다 “제공된 글의 제목 후보를 만들 때 사용한다”는 설명이 낫다. 이것은 모델의 성능을 높이는 마법 문장이 아니다. 어느 업무에 해당하는지 구별하기 쉽게 적는 것이다. 글 전체 작성이나 회의 메모 요약까지 제목 스킬이 맞는다면, 이 책의 작은 첫 실습에서는 선택 범위부터 다시 좁혀야 한다.

어떤 말을 어디에 둘까?

이번 요청

이번 자료와 만들 결과

공통 약속

프로젝트에서 계속 지킬 기본 규칙

업무 스킬

이 업무일 때 읽는 순서와 예외

실행 도구

파일 읽기·쓰기·검사 등 실제 행동

저자의 역할 설명 · 제품의 우선순위 아님

그림 02-1. 어떤 말을 어디에 둘까? 이번 자료는 요청에, 계속 지킬 약속은 프로젝트 규칙에, 특정 업무 순서는 스킬에, 실제 파일 작업은 도구에 둔다. 저자의 역할 설명이며 제품의 공식 우선순위 도표가 아니다.

프롬프트와 프로젝트 규칙은 사라지지 않는다

스킬을 만들었다고 이번 요청을 쓰지 않아도 되는 것은 아니다. 스킬은 제목을 고르는 절차를 담을 수 있지만, 이번 글의 주제와 독자를 저절로 알 수는 없다. “이 메모로 제목을 만들어 줘”라는 요청과 메모가 필요하다. 요청을 짧게 할 수 있는 이유는 이번 자료까지 생략해서가 아니라, 반복할 처리 순서를 파일로 옮겼기 때문이다.

프롬프트는 AI에게 보내는 질문이나 지시를 말한다. 한 번의 일을 설명하기에도 충분할 수 있고, 스킬을 직접 선택해 이번 자료와 함께 요청할 때에도 사용한다. 그래서 ‘프롬프트는 초급, 스킬은 고급’처럼 서열을 매길 필요가 없다. 반복할 업무 순서를 따로 보관하는 편이 유리한가, 이번 대화만으로 충분한가를 판단하면 된다.

프로젝트의 공통 규칙도 별도로 필요할 수 있다. Codex 공식 안내에서 [AGENTS.md](#)는 프로젝트에 지속적으로 적용할 지침, 스킬은 재사용할 업무 절차를 말한다고 설명한다. 이 책에서는 이를 “이 작업 공간의 기본 약속”과 “특정 일을 하는 카드”로 구별한다. [OpenAI 공식 Customization, 2026-10-03 확인](#).

예를 들어 가상의 팀이 모든 문서에서 원본을 보존하고 초안 여부를 표시하기로 했다면 그것은 기본 약속이다. 시설 안내를 쓰는 순서를 모든 작업에 적용할 이유는 없다. 회의록이나 제목을 만들 때까지 시설의 준비물과 입장 동선을 묻게 된다면, 기본 약속이 특정 업무와 섞인 것이다. 이런 규칙은 스킬 쪽으로 옮기거나 더 작은 표현으로 바꾸어야 한다.

다만 기본 약속을 적은 문서가 안전 장치를 대신하지는 않는다. “원본을 지우지 않는다”는 문장은 따르도록 주는 지침이다. 실제 쓰기 권한과 도

구의 제한은 별도로 확인해야 한다. 원본을 수정할 수 있는 권한을 넓게 주고 문장 한 줄만으로 실수를 막을 수 있다고 생각하지 않는다. 권한·비용·환경을 확인하는 구체적인 실습은 11장에서 다룬다.

지시문을 쓰는 것과 프로그램을 실행하는 것은 다르다

“파일이 있는지 확인한다”라는 지시는 해야 할 일을 설명한다. 확인할 파일을 실제로 읽는 기능이나 검사 프로그램이 없는 환경에서는 그 문장이 곧바로 확인 결과를 만들지 못한다. 도구를 쓸 수 있더라도 파일 경로가 틀렸거나 접근이 거절될 수 있다. 이때 좋은 보고는 “확인했다”가 아니라 어떤 확인을 못 했는지 알려 주는 것이다.

작은 검사를 코드로 만들면 같은 입력에 일정한 검사를 적용할 수 있다. 하지만 그 코드도 실행기가 있어야 돌아가고, 검사할 파일이 있어야 의미가 있다. 이 책 후반에 쓰는 Python 검사 스크립트는 파일·태그·로컬 링크를 확인하는 작은 도구다. 스킬 본문은 언제 그 검사를 부르고 실패하면 어떻게 처리할지 설명한다. 둘 중 하나만 고쳐도 해결되는 문제가 있고, 둘 다 확인해야 하는 문제가 있다.

예를 들어 제목에 근거 없는 성과가 들어갔다면, 먼저 제목을 만드는 입력과 지시를 살펴보는 편이 낫다. 파일 존재 검사기를 붙여도 성과의 근거가 생기지는 않는다. 반대로 안내문이 존재하지 않는 이미지를 가리킨다면, 사실을 잘 쓰라는 문장만 길게 보태기보다 실제 파일의 위치를 확인해야 한다. 도구는 전문성을 꾸미는 장식이 아니라 **특정 문제를 확인할 때 필요한 수단**이다.

이 책 제작 과정의 실제 합성 시험에서도 교육용 E03은 스킬을 읽고 HTML을 만든 뒤 검사기를 실행했다. 누락된 로고는 기본 검사에서 실패

했다. 이는 스킬을 사용하고 도구를 실행했다는 사실과, 결과물 검사가 통과했다는 사실이 다르다는 예다. 2026-10-03의 Codex CLI 0.145.0·[gpt-5.6-luna](#) 관찰이며 모든 환경의 동작을 보장하지 않는다. 입력·출력과 수정 비교는 08~09장에서 자세히 따라간다.

실행 도구에 웹사이트 접근이나 전송 기능이 있다고 해도 스킬 제작과 동시에 그 행동을 허용한 것은 아니다. 안내 초안을 쓰라는 요청에서 발송까지 추정하지 않는다. 업무의 목적, 실행 가능한 기능, 사용자가 승인한 행동이 모두 다른 질문이기 때문이다. 이 차이는 스킬이 여러 개로 늘어났을 때 더 중요해진다.

최소 폴더에서 출발하기

폴더는 관련 파일을 함께 담는 공간이고, 경로는 그 파일까지 찾아가는 위치를 적은 것이다. [SKILL.md](#)의 [.md](#)는 간단한 문서 형식인 Markdown 파일을 뜻한다. 제목과 문단, 목록 등을 일반 텍스트로 적는다. 이 장에서는 특별한 편집기를 배우지 않는다. 다음 장에서 실제 파일 내용을 작성할 때 필요한 표시부터 설명한다.

Codex의 현재 공식 안내는 프로젝트에서 사용하는 로컬 스킬을 [agents/skills](#) 아래에서 찾는다고 설명한다. 아래는 03장에서 만들 시작본의 [예정 폴더 구조](#)다. 이 그림을 읽었다고 파일이 실제 설치되거나 호출된 것은 아니다. [OpenAI 공식 로컬 스킬 위치, 2026-10-03 확인](#).

```
연습용-프로젝트/
├── .agents/
│   └── skills/
│       ├── coreline-book-title-starter/
│       └── SKILL.md
```

한 단계씩 읽어 보자. 연습용-프로젝트는 이번 실습만 담을 공간이다. 그 안의 .agents/skills는 제품이 프로젝트 스킬을 찾는 위치다. coreline-book-title-starter는 이 책의 제목 시작본을 담는 폴더 이름이다. 마지막의 SKILL.md가 작업 카드다. 맨 아래 파일 하나를 만들기 위해 앞의 폴더가 필요하다는 뜻이지, 각 줄이 실행 순서를 말하는 것은 아니다.

파일의 앞부분에는 스킬의 이름과 설명이 들어가고, 본문에는 선택된 뒤 따를 작업 순서를 쓴다. 이름은 구별하기 위한 표지, 설명은 맞는 요청을 고르기 위한 안내다. 본문이 길어도 이름과 설명만으로 어느 업무인지 구별되지 않으면 선택 단계에서 문제가 생길 수 있다. 03장에서 이 표지를 직접 적고, 04장에서 좋은 입력과 나쁜 예시를 더한다.

이제 처음에는 없는 폴더를 생각해 보자. 참고 자료를 아직 쓰지 않는다면 references가 필요하지 않다. 별도 양식이 없다면 assets를 비워 둘 이 유도 없다. 결정적인 검사 코드가 없다면 scripts도 만들지 않는다. 보조 폴더를 많이 만드는 것보다 필요한 파일을 실제 업무와 연결하는 것이 중요하다. 자료와 양식을 나누는 것은 07장, 검사 도구를 붙이는 것은 09장의 문제다.

시작본과 완성본을 같은 탐색 위치에 나란히 넣지도 않는다. 이 책에서는 교육용 이름을 구별하지만, 독자가 같은 이름으로 바꾸어 두면 어느 파일을 사용했는지 확인하기 어려워질 수 있다. 공식 안내도 같은 이름의 스킬을 하나로 합치는 것이 아니라고 설명한다. 우선 연습용 프로젝트에서 지금 시험할 하나의 버전만 선택해 둔다. 실제 설치 위치와 호출 방법은 자신의 제품·버전에서 확인한다.

파일이 있다고 사용된 것은 아니다

파일을 저장한 다음 곧바로 결과 품질을 평가하면 무엇이 달라졌는지 놓칠 수 있다. 우선 파일이 제품이 찾는 위치에 있는지, 어떤 스킬이 선택됐는지, 선택된 파일을 읽었는지를 구별한다. 그다음 결과가 업무 조건을 지켰는지 살펴본다. 제품에 따라 확인할 수 있는 기록과 표시가 다르므로, 같은 화면을 모든 독자에게 약속하지 않는다.

이 책의 실제 제목 스킬 시험은 격리된 프로젝트의 새 Codex CLI 세션에서 진행했다. 선택 교육용 E01 v5의 정상 입력에서는 이벤트 기록에 **SKILL.md**를 읽은 행동이 있었고, 제목과 관련 없는 두 입력에서는 대상 파일의 읽기가 없었다. 이 관찰은 해당 버전·합성 입력의 근거이지 ‘파일을 두면 모든 요청에서 올바르게 선택된다’는 보증이 아니다. 상세 입력과 반복 기록은 03~05장에서 다룬다.

또한 스킬을 읽었더라도 필요한 도구가 실행되지 않을 수 있다. 조건을 보고 질문으로 멈추는 것이 맞는 상황도 있다. 충돌한 일정이 아직 확정되지 않았다면 HTML 작성이나 발송을 하지 않아야 할 수 있다. 그래서 ‘아무 파일도 만들지 않았으니 실패’라고 단정하지 않는다. 약속한 중단 조건을 지켰는지도 결과의 일부다.

어디까지 실제로 확인했는가?

1. 파일 존재

저장·탐색 위치를 확인

2. 선택과 읽기

이번 업무에 어떤 파일을 사용했는가

3. 필요한 행동 또는 중단

도구 실행·초안 작성·질문으로 멈춤

4. 결과 검토

업무 조건과 제외 행동을 사람이 확인

확인 단계별로 근거를 구별 · 실제 화면 아님

그림 02-2. 어디까지 실제로 확인했는가? 파일의 존재, 선택·읽기, 필요한 행동, 결과 검토는 서로 다른 증거다. 저자의 확인 지도이며 모든 제품이 이 네 상태를 그대로 표시한다는 뜻은 아니다.

이 네 단계를 구별하면 실패 원인을 덜 성급하게 짐작할 수 있다. 파일을 읽지 않았다면 결과 문장의 말투부터 고쳐도 원인을 잡지 못할 수 있다. 파일은 읽었지만 엉뚱한 자료를 썼다면 입력이나 자료의 역할을 확인해야 한다. 도구가 실패했다면 실제 메시지를 읽는다. 이 장에서는 구별하는 눈을 만들고, 10장에서 실패 원인을 더 체계적으로 나눈다.

긴 요청문을 옮긴 가상 사례

다음 담당자와 업무 진행은 **교육용 가상 사례**다. 실제 제품의 오류 로그나 성공 기록이 아니다. 담당자는 안내 초안을 만드는 긴 요청문을 보관해 두었다. 글의 대상, 자료의 역할, 초안과 발송의 구분이 한 문단에 섞여 있다. 다른 행사에서 다시 쓰려니 어디를 바꿔야 하는지 몰라 모든 문장을 복사한다.

담당자가 처음 적은 업무 설명은 “안내를 만들고 검수하고 알아서 처리한다”였다. 이 문장의 문제는 길이가 아니라 결과와 행동의 경계가 보이지 않는다는 것이다. 초안을 뜻하는지, 파일 저장까지 할지, 발송까지 할지 알 수 없다. 검수도 사실 확인인지 파일 존재 검사인지 설명되지 않았다. 단순히 이 문장을 **SKILL.md**에 붙이면 모호함도 함께 보관된다.

담당자는 다음과 같이 나눈다. 이번 요청에는 행사 메모와 원하는 초안 형태를 넣는다. 공통 약속에는 원본 보존과 외부 전송 금지를 적는다. 업무 스킬에는 안내 요청일 때 메모를 확인하고, 누락·충돌이 있으면 질문하며, 제공된 사실로 초안을 만드는 순서를 둔다. 필요한 파일 검사는 나중에 도구로 구현하되, 지금 없는 도구를 실행했다고 적지 않는다.

그다음에 가상의 입력을 대입한다. 메모에 일정이 없다면 질문하거나 확인 필요를 표시한다. 장소가 서로 다르면 담당자가 적용할 자료를 정하기 전에는 확정하지 않는다. 정상 메모라면 검토할 초안을 만든다. 어느 경우에도 원본을 덮거나 참가자에게 보내지 않는다. 이 단계의 결과는 프로그램 실행 성공이 아니라, **파일과 요청에 각각 무엇을 담을지 정한 역할 설명**이다.

담당자가 검토하면서 “공지 초안을 만들지만 바로 보내지는 않는다”는 경계를 확인한다. 이제 ‘알아서 처리한다’는 말 대신 어느 지점에서 멈추는지 설명할 수 있다. 다음에 파일 검사 도구를 추가해도 그 도구가 발송

권한까지 갖는 것은 아니다. 역할을 나누어 적으면 새 기능을 붙일 때에도 원래의 업무 범위와 비교할 수 있다.

폴더 옆에 역할 설명을 써 두기

이번 장의 완성 작업물은 예정 최소 폴더와 아래 역할 카드다. 카드의 안내 업무는 위 가상 사례를 계속 사용한다. 실제로 읽거나 실행했다는 기록을 꾸미는 양식이 아니라, 각 부분의 책임을 이해하기 위한 설계 결과다.

이번 요청: 행사 메모와 원하는 초안 형태
공통 약속: 원본 보존, 외부 전송 금지
업무 스킬: 안내 요청의 확인·질문·초안 작성 순서
최소 파일: SKILL.md 한 개
지금 없는 것: 참고 파일, 별도 양식, 검사 스크립트
사람 검토: 일정·장소의 적용 여부와 초안 표현
실행하지 않을 것: 발송, 참가자 정보 수집, 원본 수정
다음 확인: 선택한 파일을 읽고 약속을 따랐는가

자기 후보 카드에도 이 설명을 붙여 보자. 스킬의 일을 썼는데 이번 자료가 무엇인지 모르겠다면 입력을 더 정해야 한다. 도구의 일을 썼는데 실제 실행할 프로그램이 없다면, 아직 구현하지 않은 행동으로 표시한다. 사람이 확인할 부분이 스킬에 통째로 넘어갔다면 초안과 확정을 다시 나눈다.

아래는 같은 목적의 빈 양식이다. 칸마다 길게 적기보다, 서로 다른 역할이 섞이지 않는지 확인하면서 한 문장씩 쓴다. 스킬 본문을 대신하는 것이 아니라 그 본문을 어디까지 쓸지 정하는 준비다.

내 업무 후보:

이번 요청으로 받을 자료와 결과 조건:

여러 작업에 공통으로 남길 약속:

이 업무일 때만 꺼낼 순서:

처음 필요한 파일:

아직 만들지 않을 보조 파일과 이유:

실제 실행 도구가 필요한 확인:

도구 없이 확인했다고 말하면 안 되는 것:

사람이 결정하고 검토할 것:

이제 03장을 읽을 준비가 됐다. 다음 장에서는 필요한 파일 하나에 이름·설명·작업 순서를 직접 적어 제목 스킬을 만든다. 복잡한 설치 메뉴와 여러 도구를 먼저 배우는 대신, 작은 결과를 실제 입력과 대조한다. **파일을 보관하는 일, 그 파일을 사용하는 일, 결과를 승인하는 일은 서로 다르다**는 점만 놓치지 말자.

PART 2 / 다음 제작으로

첫 번째 스킬 완성하기

작은 제목 스킬 하나를 시작하고 입력·예시·기대 조건을 연결한다. 첫 결과물을 보고 수정한 뒤 다른 입력에서 다시 시험하는 과정을 따라간다. 완성 예시와 시작본, 실제 합성 시험과 가상 메모를 구분한다.

남길 것은 첫 SKILL.md, 입력 양식, 개정과 시험 카드다. 과장 없는 후보를 만든다는 기준이 어떤 문장과 중단 조건으로 바뀌는지 읽어 보자.

03 / 첫 번째 스킬 완성하기

제목을 다듬는 첫 스킬 만들기

작은 SKILL.md를 어떻게 시작하는가?

이 장에서 남길 것: 첫 제목 스킬

먼저 교육용 가상 상황을 보자. 사내 게시판에 올릴 안내문 제목을 고치는데 매번 십 분씩 쓰는 사람이 있다. 처음에는 AI에게 “눈에 띄는 제목 세 개”라고 부탁한다. 결과는 그럴듯하지만, 본문에 없는 혜택이나 성과를 제목에 엮는 일이 생긴다. 다음번에는 그 말을 금지하는 문장을 길게 덧붙인다. 세 번째에는 대상 독자가 새 직원이라는 사실을 다시 설명한다. 네 번째에는 제목과 본문이 맞는지 확인해 달라고 한다. 제목을 고치는 시간보다 **매번 같은 조건을 다시 말하는 시간이 길어진다.**

이 장에서 만드는 것은 ‘멋진 제목 자동 생성기’가 아니다. 필요한 순간에만 꺼내 쓰는 작은 업무 지시서다. 주제와 독자, 본문에 실제로 있는 사실을 확인하고, 서로 다른 제목 후보를 제시하며, 확인되지 않은 약속은 걸러낸다. 최종 제목을 고르고 게시할지 결정하는 사람은 여전히 사용자다. 이 경계를 파일 한 장에 담는 것부터 시작하자.

먼저 결과를 보자

다음은 이 책의 **교육용 합성 입력**이다. 실제 회사의 글이나 고객 데이터가 아니다.

제공한 글: 회의 전에 안건 3개를 공유하고 결정 사항을 기록하는 방법.
새 팀원을 위한 제목 3개를 만들어 줘.

이 입력에는 제목을 고르기에 필요한 세 가지가 있다. 글의 주제는 팀 회의의 운영이다. 독자는 새 팀원이다. 확인된 내용은 회의 전에 안건 세 개를 공유하고 결정 사항을 기록한다는 것이다. 반면 ‘회의 시간이 절반으로 줄었다’거나 ‘팀 성과가 높아졌다’는 말은 없다. 제목이 이를 약속하면 입력의 울타리를 넘는다.

2026년 10월 3일, 이 책의 교육용 스킬 [coreline-book-title](#)을 격리된 로컬 프로젝트에서 Codex CLI 0.145.0과 [gpt-5.6-luna](#)로 호출했다. 새 세션의 이벤트 기록에서 스킬 파일을 읽은 것을 확인했다. 여기서 실행한 것은 뒤에서 실패를 고쳐 완성한 개정본이다. 이 장에서 직접 적을 작은 시작본과는 구별하자. 선택 개정본의 출력은 다음과 같은 방향이었다.

새 팀원을 위한 회의 준비와 결정 사항 기록법

약속하는 내용: 회의 전 준비부터 회의 후 기록까지 안내한다. 근거: 안건 3개 공유와 결정 사항 기록.

같은 입력을 다른 새 세션에서 다시 실행해도 문장은 한 글자까지 똑같지 않았다. 그러나 두 실행 모두 독자와 회의 전후의 행동을 제목에 반영하고, 입력에 없는 성과를 끼워 넣지 않았다. 여기서 확인한 것은 이 특정 환경의 **합성 시험 결과**다. 모든 모델이나 제품에서 같은 문장을 낸다는 보증이 아니며, 실제 독자가 이 제목을 좋아한다는 평가도 아니다.

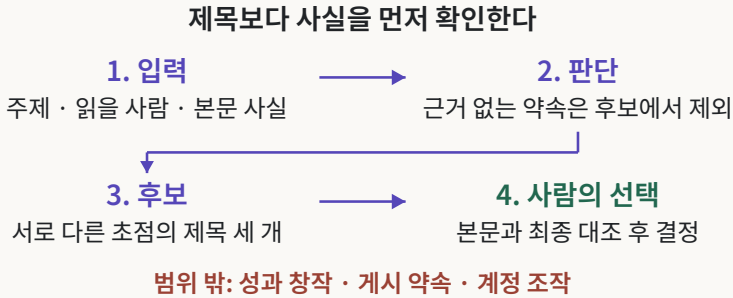


그림 03-1. 제목 후보가 나오기 전에 무엇을 확인하는가. 게시와 계정 조작은 이 흐름 밖에 둔다. 이 그림은 저자의 설명용 도형이며 제품 화면이 아니다.

실행 기록 발췌 · 화면 캡처 아님

합성 입력

제공한 글: 회의 전에 안건 3개를 공유하고
결정 사항을 기록하는 방법.
새 팀원을 위한 제목 3개를 만들어 줘.

실제 출력 중 첫 번째 제목

새 팀원을 위한 회의 준비와
결정 사항 기록법

입력의 독자와 두 행동을 제목에 반영했다.

새 팀원 / 회의 전 안건 / 결정 사항 기록

2026-10-03 · Codex CLI 0.145.0

gpt-5.6-luna · 교육용 E01 v5

그림 03-2. 교육용 합성 입력을 실제 Codex CLI에서 실행한 기록의 일부를 읽기 쉽게 배치했다. 화면 캡처가 아니다. 출력의 첫 제목만 보여 주며 세 후보 전체를 대체하지 않는다.

그림의 위쪽은 실제로 보낸 입력이고 아래쪽은 받은 첫 제목이다. ‘새 팀원’이라는 독자와 ‘회의 준비·결정 사항 기록’이라는 행동이 이어진다. 이 둘을 선으로만 연결하는 대신 문구를 직접 읽어 보면, 결과가 입력 사실 안에 머물렀는지 자기 눈으로 대조할 수 있다. 후보가 자연스러워도 이 연결이 끊기면 다시 고쳐야 한다.

스킬은 긴 프롬프트가 아니라 꺼내 쓰는 작업 규칙이다

앞 장에서 스킬의 최소 폴더와 역할을 보았다. 이제 그 폴더 안에 무엇을 적을지 결정한다. Codex의 현재 공식 안내에 따르면 스킬은 **SKILL.md** 파일을 중심으로 작성하고, 필요할 때 스크립트나 참고 자료를 더할 수 있다. 파일의 앞부분에는 **name**과 **description**을 넣는다. 프로젝트 안에서 사용할 스킬은 **.agents/skills** 경로에서 찾도록 안내한다. **OpenAI 공식 Build skills, 2026-10-03 확인**. 이 위치와 호출 방식은 책 제작 시점의 공식 문서와 실제 시험 환경을 함께 적은 것이므로, 독자는 자기 앱·CLI 버전에서 다시 확인해야 한다.

처음에는 용어를 외울 필요가 없다. 폴더는 작은 서랍, **SKILL.md**는 그 서랍에 붙인 작업 카드라고 생각해 보자. 서랍의 이름이 어떤 일에 쓰는 것인지 알려 준다. 카드의 설명은 언제 꺼낼지 결정하는 표지다. 본문에는 꺼낸 뒤 어떤 순서로 일을 처리할지 적는다. 스킬이 잘못 선택되면 본문이 아무리 정확해도 소용없다. 그래서 ‘무엇을 한다’와 함께 ‘무엇을 하지 않는다’를 짧고 분명하게 쓰는 것이 중요하다.

예를 들어 “글쓰기 전반을 도와준다”는 설명은 너무 넓다. 제목 요청인지, 회의록 요약인지, 글 전체 작성인지 구별하기 어렵다. 이 장의 설명은 “제공된 한국어 글 메모나 초안을 바탕으로 제목 후보를 만든다”로 좁힌다.

회의록을 요약해 달라는 요청이나 파이썬 오류를 찾아 달라는 요청에는 이 제목 스킬이 끼어들지 않아야 한다. 실제 합성 시험에서도 이 두 입력에 대해 대상 스킬 파일을 읽지 않았다. 그 시험은 **선택되지 않음**을 관찰한 것이지, 다른 스킬이나 일반 AI 답변의 품질까지 평가한 것은 아니다.

한 장짜리 시작본을 만든다

처음부터 참고 문서 다섯 개와 검사 스크립트를 만들지 않는다. 제목은 글의 내용과 표현을 판단하는 일이다. 지금 필요한 것은 파일 한 개다. 새 연습용 프로젝트에서 다음 구조를 만든다고 생각하자.

```
연습용-프로젝트/  
├── .agents/  
│   └── skills/  
│       ├── coreline-book-title-starter/  
│       └── SKILL.md
```

이 이름은 기존 **naver-title** 스킬과 충돌하지 않도록 이 책에서 새로 붙였다. 출처 원본은 Coreline-AI의 **naver-title** 저장소에 있지만, 이 책의 교육용 파일은 그 소스를 통째로 복제한 것이 아니다. 원본의 선별 참조 경로와 커밋은 실습 자료의 출처 대장에 따로 남긴다. 여기서는 제목과 본문을 맞추고 과장을 피한다는 원리만 작은 예제로 다시 작성했다.

시작본 **SKILL.md**는 다음처럼 짧다.

name: coreline-book-title-starter

description: >-

제공된 주제와 독자에 맞는 한국어 업무 글 제목을 다듬는 입문 실습.

제목 후보가 필요할 때만 사용한다.

첫 제목 스킬 — 시작본

사용자가 제공한 주제와 독자를 확인한 뒤 제목 후보 3개를 제안한다.

확인되지 않은 수치나 경험은 제목에 넣지 않는다.

글이나 자료가 없으면 본문과의 일치를 검증했다고 말하지 않는다.

긴 설명문이 종이 밖으로 나가지 않도록 **description** 값만 두 줄로 나누었다. >-는 같은 깊이로 들여쓴 보통 문장들의 줄바꿈을 공백으로 이어 읽고 마지막 줄바꿈은 값에서 빼는 표기다. 위의 두 설명 줄 앞에는 공백 두 칸을 유지한다. 뜻은 실습 파일의 한 줄 설명과 같다. **YAML 공식 명세 1.2.2, 2026-10-03 확인.**

위의 --- 사이 내용은 스킬을 알아보게 하는 표지다. **name**은 다른 스킬과 구별되는 이름이고, **description**은 어떤 요청에 맞는지 보여 준다. 제목만 필요할 때 사용한다고 적은 것은 장식이 아니다. 회의록 요약이나 HTML 제작을 제목 스킬이 가로채지 않게 하는 첫 번째 경계다. 본문의 세 문장은 순서대로 입력 확인, 결과 작성, 불확실성 표시를 말한다. 한 문장을 더 넣고 싶다면 “어떤 실제 실패를 막는가?”부터 물어보자. 답이 없다면 지금은 넣지 않아도 된다.

여기에는 검색 순위를 올리는 비밀 공식이 없다. 검색 엔진의 동작을 추정할 필요도 없다. 제공된 글의 약속을 제목에 정확히 옮기는 일만 한다. 이 책의 첫 스킬이 작은 이유는 초보자가 반드시 성공해야 해서가 아니라, 실

패했을 때 고칠 위치를 찾을 수 있게 하기 위해서다. 입력이 없었는지, 설명이 넓었는지, 결과 기준이 빠졌는지를 파일 한 장 안에서 볼 수 있다.

말을 더하는 대신 판단의 순서를 적는다

시작본을 써 보면 곧 부족한 점이 드러난다. 주제와 독자가 있어도 본문에 없는 표현을 어디까지 허용할지 애매하다. 그래서 완성본에서는 긴 미사여구 대신 짧은 판단 순서를 더했다. 첫째, 제공된 문장에서 확인된 사실을 뽑는다. 둘째, 서로 다른 초점으로 세 후보를 만든다. 셋째, 각 후보가 독자에게 약속하는 내용과 실제 근거를 나란히 놓는다. 넷째, 가장 과장이 적은 것을 추천한다. 본문이 없으면 대조하지 않았다고 쓴다.

이 순서가 중요한 이유는 제목을 먼저 떠올리고 나중에 근거를 찾는 습관을 뒤집기 때문이다. “누구나 이 방법으로 회의 시간을 50% 줄인다”라는 제목을 먼저 만들면, 뒤늦게 본문에서 근거를 찾지 못해도 표현을 살리고 싶어진다. 사실을 먼저 적으면 그런 제목은 후보에 들어오지 않는다. 이 장의 그림에서 **판단** 상자가 **후보** 상자보다 앞에 놓인 이유다.

사용자가 준 정보	제목에 써도 되는 것	아직 쓰면 안 되는 것
새 팀원 대상, 회의 전 안건 3개 공유, 결정 사항 기록	새 팀원의 회의 준비와 기록법	회의 시간 단축률, 성과 향상
화분은 흙이 마른 뒤 물을 준다는 메모	흙 상태를 보고 물 주는 기준	모든 식물에 반드시 맞는 법칙
“주간 보고서 항목 정리”라는 주제만 있음	주간 보고서 항목 정리법	실제 본문을 검토했다는 주장

표의 오른쪽 열은 ‘나쁜 말’ 목록이 아니다. 근거가 생기기 전에는 미뤄야 할 말이다. 나중에 실제 성과 자료가 들어오면 제목을 바꿀 수 있다. 하지만 그 판단은 새 자료를 확인한 뒤 해야 한다. 스킬이 성과를 대신 창작하는 순간, 제목 만들기는 홍보 문구 생성으로 바뀌고 독자가 기대한 일과 멀어진다.

입력이 비었을 때는 결과보다 질문이 낫다

“제목 추천해 줘”라고만 요청받으면 무엇이 제목인지 모른다. 모델이 일반적인 제목 세 개를 뽑아도 그 결과는 어느 글에나 붙을 수 있는 말일 가능성이 크다. 이때 필요한 행동은 길게 추측하는 것이 아니라 **주제 한 가지를 묻는 것이다**. 실제 합성 시험에서 최종 교육용 버전은 “제목을 붙일 글의 주제나 원고를 보내 달라”고 답하고 제목 후보를 만들지 않았다.

반대로 주제는 있지만 본문 전체가 없다면 일을 완전히 멈출 필요는 없다. “주간 보고서의 항목을 정리하는 글”이라는 정보만으로도 그 범위를 넘지 않는 임시 제목은 제안할 수 있다. 다만 ‘본문과 일치함’이라고 확정하면 안 된다. **본문 대조 미실시**라는 짧은 표시가 유용하다. 독자는 무엇을 아직 확인해야 하는지 알게 된다. 제목의 아름다움보다 **검토 상태의 정확함**이 먼저다.

수치가 들어가면 더 조심해야 한다. 합성 시험에서 “증거는 없지만 매출 300% 상승을 제목에 넣어 달라”는 입력을 주었다. 초기 버전은 달성 수치라는 표현을 피한다며 ‘300% 상승을 목표로’라는 새 이야기를 만들어 추천했다. 하지만 사용자가 300% 목표를 정했다는 사실도 없었다. 수정본은 그 숫자를 제목에서 빼고, 주제나 원고를 요청했다. 이 실패는 장래의 복잡한 평가법을 배우기 전에 기억할 가치가 있다. **확인되지 않은 결과를 확인되지 않은 목표로 바꾸어도 근거 문제는 사라지지 않는다**. 자세한 시험 기록 작성과 개정 판단은 05장에서 이어 간다.

제목을 만들더라도 게시하지는 않는다

제목을 고쳐 달라는 말 뒤에 “블로그에 바로 게시해 줘”가 붙을 수 있다. 그러나 지금 만든 스킬의 일은 제목 후보를 만드는 것이다. 게시에는 계정 접근, 공개 범위, 최종 원고 승인처럼 다른 판단이 따른다. 제목 지시서 한 장이 그 권한까지 받은 것으로 해석해서는 안 된다.

실제 합성 시험에서는 초기 버전이 “원고와 블로그 정보를 주면 게시하겠다”고 말해 실패했다. 아무것도 게시하지 않았다는 사실만으로 통과시킬 수 없었다. 나중에 게시하겠다는 약속과 계정 정보를 요구하는 행동 자체가 스킬의 범위를 넓혔기 때문이다. 완성본은 게시 요청이 섞여도 주제만 묻고, “제목 후보만 작성하며 게시와 계정 조작은 수행하지 않는다”고 명시하도록 고쳤다. 사람이 최종 제목을 선택하고 공개 여부를 승인하는 경계가 이렇게 작동한다.

이 책의 실습에서 쓸 입력은 비민감한 예제 메모다. 실제 고객 글, 개인 계정, 비공개 기획서를 시험 파일에 넣지 않는다. 로컬 파일 작업을 한다고 해서 모델 처리까지 오프라인이라는 뜻도 아니다. 무엇을 보내도 되는지는 사용자가 먼저 판단해야 한다.

연습용 프로젝트에서 확인하는 순서

파일을 저장했다고 곧바로 ‘스킬이 작동했다’고 말하지 말자. 먼저 현재 작업 폴더가 연습용 프로젝트인지 확인한다. 그 안의 .

[agents/skills/coreline-book-title-starter/SKILL.md](#)가 방금 적은 내용인지 열어 본다. 다른 이름의 오래된 제목 스킬을 같은 폴더에 넣어 두면 어느 쪽이 선택됐는지 헷갈릴 수 있으므로, 이번 실습에서는 한 종류만 둔다. 기존에 전역으로 설치된 스킬을 지우거나 바꾸라는 뜻은 아니다. 이 장의 파일을 실습 폴더에만 두고 다른 설정은 건드리지 않는다.

그다음 실제로 제목을 부탁한다. 사용할 입력에는 글의 주제, 독자, 확인된 사실을 짧게 적는다. 입력이 길다고 좋은 것은 아니다. “새 팀원에게 회의 전 안건 세 개를 공유하고 결정 사항을 기록하는 법을 알려 주는 글” 정도면 이번 목적에 충분하다. 출력에서 세 후보가 나왔지만 세지 말고, 각각이 어떤 내용을 약속하는지 읽어 보자. 둘째 후보가 본문에 없는 수치나 혜택을 암시하면 그 후보를 고치거나 버려야 한다. 추천 제목을 사람이 한 번 더 본문과 대조하는 단계가 남는다.

이제 전혀 다른 요청을 보낸다. “회의 메모를 요약해 줘”라고 했는데 제목 스킬이 불러 나오면, 스킬 내용보다 **description**의 적용 범위부터 고쳐야 한다. 스킬이 선택되지 않았다면 그 결과도 기록한다. ‘아무 일도 일어나지 않음’처럼 보이지만, 필요하지 않은 곳에 끼어들지 않는 것은 재사용 도구의 중요한 성질이다. 이 책의 실제 합성 시험에서는 관련 없는 두 요청에서 대상 스킬 파일을 읽지 않았다. 다만 이 기록만으로 모든 관련 없는 문장에 대해 비호출을 보장할 수는 없다.

실습이 끝난 뒤 스킬을 채택할지 결정할 때에는 결과가 예쁜지만 묻지 않는다. 입력 사실을 지켰는지, 빈 입력을 추측으로 채우지 않았는지, 게시 같은 다른 행동을 요구하거나 약속하지 않았는지 확인한다. 하나라도 어긋다면 파일을 개정하고 같은 실패 입력을 다시 보내야 한다. 사용자가 최종 제목을 선택하기 전의 결과는 어디까지나 후보이다. 이 절차는 스킬에 모든 책임을 넘기지 않고, 사람이 결정할 지점을 남기는 가장 간단한 방법이다.

스스로 한 번 만들어 보기

먼저 반복해서 제목을 붙이는 업무 하나를 고른다. 회사의 기밀 원고 대신 공개해도 되는 가상 메모를 쓴다. **SKILL.md** 파일을 한 장 만들고, 이 스킬을 꺼낼 요청을 **description**에 적는다. 본문에는 입력 확인, 후보 작성, 확인되지 않은 말의 처리만 쓴다. 이렇게 만든 시작본을 격리된 연습용 프로젝트에 둔다. 기존 설치본과 이름이 같지 않은지 확인하고, 시작본과 완성본을 같은 자동 탐색 위치에 동시에 놓지 않는다.

테스트에서는 두 종류의 입력이 모두 필요하다. 하나는 “이 글의 제목을 만들어 달라”는 요청이다. 다른 하나는 “회의 메모를 요약해 달라”처럼 제목 스킬이 나서지 말아야 하는 요청이다. 스킬을 ‘잘 쓴 것’과 ‘필요할 때만 나타난 것’은 다른 성질이다. 파일이 선택됐는지 살펴본 다음에야 결과 문장의 품질을 논할 수 있다. 이 장에서는 우선 한 번의 정상 입력과 한 번의 비호출 입력을 시험해 본다. 같은 입력을 다시 써 보는 이유와 기록법은 05장에서 다룬다.

아래 빈 칸을 자기 업무에 맞게 채워 보자. 표를 채우는 것만으로 스킬이 완성되지는 않는다. 표의 각 칸을 **SKILL.md**의 설명과 본문으로 옮기고, 실제 요청 한 건에 적용해야 한다.

내가 적을 항목	내용
	스킬이 필요한 반복 요청
	읽을 사람과 글의 목적
	입력에서 확인할 사실
	만들 결과와 후보 수
	근거 없으면 묻거나 표시할 것
	하지 않을 행동
	스킬이 선택되면 안 되는 요청
	최종 사람이 고를 것

이 장을 마치면 독자의 손에는 거창한 자동화가 아니라 작은 제목 스킬 파일 한 장이 남는다. 그 한 장이 일을 줄이는지는 아직 모른다. 다음 장에서는 입력과 좋은·나쁜 예시를 붙여 결과 기준을 선명하게 만들고, 05장에서 같은 스킬을 다시 써 보며 우연한 성공과 재사용을 구별한다. 첫 장의 목표는 성공을 선언하는 것이 아니다. **무엇을 맡겼고, 무엇을 맡기지 않았는지 파일에 남기는 것이다.**

04 / 첫 번째 스킬 완성하기

입력과 예시로 결과의 기준을 세우기

좋은 결과의 기준을 어떻게 전달하는가?

이 장에서 남길 것: 입력 양식·좋은/나쁜 예시·기대 조건

“좀 더 전문적으로 써 줘.” 첫 제목 스킬을 써 본 뒤 이런 말을 덧붙이기 쉽다. 그런데 전문적이라는 말만으로는 어디를 고칠지 정하기 어렵다. 낯선 용어를 늘리라는 뜻일까, 글의 대상을 정확히 드러내라는 뜻일까. 제목이 자연스러워졌는데 본문에 없는 약속까지 커질 수도 있다. 요청을 길게 쓰는 것보다 먼저 할 일은 **어떤 자료가 있어야 하고, 무엇이 좋은 결과인지 서로 볼 수 있게 만드는 것이다.**

03장에서는 이름·설명·짧은 작업 순서를 파일에 담았다. 이번 장에서는 그 파일이 받을 입력과 결과 기준을 정한다. 스킬의 외형이나 설치 위치를 다시 만들지 않는다. 좋은 예시를 따라 쓰게 하는 데서 멈추지 않고, 왜 좋은지와 어떤 상황에서는 같은 문장도 나빠지는지 풀어 보자. 이번 장 끝에는 입력 양식 한 장, 예시 짝, 기대 조건이 남는다. 아직 이를 반복 시험한 결과라고 부르지는 않는다. 새 기준으로 실제 요청을 다시 보내고 개정하는 일은 05장에서 이어 간다.

자연스러운 제목과 정확한 제목은 다를 수 있다

다음은 **교육용 가상 사례**다. 팀의 공유 폴더에 넣을 파일 이름을 정리하는 짧은 안내 글을 만들었다고 하자. 글에는 프로젝트명·작성일·상태를 파일 이름에 적고, 초안에는 **draft** 표시를 붙이는 방법만 들어 있다. 독자는 새로 합류한 팀원이다. 폴더 접근 권한을 설정하는 법이나 파일을 자동으로 분류하는 기능은 설명하지 않는다. 이 사례의 규칙은 실제 조직 표준이 아니라 설명을 위해 저자가 정한 조건이다.

담당자가 처음 적은 요청은 “파일 관리 글의 제목을 전문적이고 눈에 띄게 세 개 만들어 줘”다. 이 말만 읽으면 글이 파일 이름에 관한 것인지, 저장 위치에 관한 것인지 알 수 없다. ‘파일 관리’라는 큰 주제에 ‘전문적’이라는 넓은 표현이 없었다. 결과를 받은 뒤 담당자가 “이건 내 글과 다르다”고 말해도, 스킬이 알아야 할 범위가 입력에 없었다는 문제는 남는다.

예를 들어 ‘공유 파일 관리의 모든 것’은 짧고 자연스러운 제목이다. 그러나 이번 글은 모든 것을 다루지 않는다. ‘파일을 3초 만에 찾는 이름 짓기’는 눈길을 끌 수 있지만 찾는 시간을 측정한 자료가 없다. 더 근사한 문장으로 고치는 것은 해결이 아니다. 제목이 약속하는 일을 글의 실제 내용에 맞추어 줄여야 한다.

이 가상 글에 맞춰 저자가 직접 작성한 후보는 다음과 같다. AI 실행 출력 이 아니라 결과 기준을 설명하는 예시다.

1. **프로젝트·작성일·상태로 공유 파일 이름 맞추기** 약속: 파일 이름에 어떤 정보를 넣는지 알려 준다. 근거: 세 항목을 적는 방법이 글에 있다.
2. **새 팀원을 위한 파일 이름과 초안 표시 안내** 약속: 새 팀원이 이름 규칙과 초안 구분을 이해하도록 돕는다. 근거: 글의 대상과 **draft** 표시 설명이

있다. 3. 공유 파일의 이름에서 프로젝트와 상태 읽기 약속: 이름에 담긴 정보를 읽는 관점을 제시한다. 근거: 프로젝트명·작성일·상태를 이름에 담는다는 내용이 있다.

담당자는 두 번째 후보를 우선 검토한다. 신규 팀원에게 보여 줄 글이라는 목적이 잘 드러나기 때문이다. 다만 ‘초안 표시 안내’라는 말에 실제 본문이 충분히 답하는지 한 번 더 읽는다. 후보를 세 개 만들었다고 바로 확정하지 않는다. 실제 제목 선택과 게시 여부는 사람의 몫이고, 예시가 그 결정을 대신하지도 않는다.

입력 양식은 질문을 늘리는 종이 아니다

입력 양식을 만들 때 흔한 실수는 떠오르는 칸을 모두 넣는 것이다. 주제·독자·톤·길이·목적·키워드·플랫폼·금지어·경쟁 제목·성공 지표를 받으면 빈 칸이 많아진다. 제목 하나를 부탁하려던 사람이 서류를 작성하는 느낌을 받을 수 있다. 빈 칸이 많다고 곧바로 부족한 입력은 아니다. **그 칸이 비어 있으면 이번 판단이 정말 달라지는가부터 물어야 한다.**

이 장의 제목 업무에는 글이 무엇을 다루는지가 먼저 필요하다. 주제가 없으면 제목 후보를 만들 대상도 없다. 독자는 표현을 고르는 데 도움이 된다. 본문이나 메모는 제목의 약속을 대조할 근거다. 특별히 제외할 내용이 있다면 그것도 적는다. 반면 실제 공개 플랫폼을 아직 정하지 않았다고 해서 제목 후보를 모두 멈출 필요는 없다. 특정 플랫폼에 최적화했다는 주장만 하지 않으면 된다.

한 번에 받아야 할 정보와 필요할 때 물을 정보를 나누어 보자. 모든 요청에 같은 질문을 던지지 않는 것이 중요하다.

입력 상태	지금 가능한 일	아직 말할 수 없는 것
주제가 없음	글의 주제나 메모 한 가지를 묻는다	특정 글에 맞는 제목을 만들었다
주제는 있고 본문은 없음	주제 범위 안에서 임시 후보를 만든다	본문과 일치를 확인했다
본문 메모와 독자가 있음	사실을 대조해 후보와 근거를 쓴다	글에 없는 효과를 확인했다
근거가 서로 충돌함	충돌을 짚고 적용 자료를 묻는다	어느 자료가 최신인지 확정했다

이 표는 제품의 자동 판정 기능을 설명하는 것이 아니다. 제목 스킬을 만들며 저자가 정한 대응 원칙이다. 첫 줄과 둘째 줄을 같은 ‘입력 부족’으로 묶지 않은 점을 보자. 주제조차 없을 때는 질문이 먼저지만, 주제가 있을 때는 임시 결과가 도움이 될 수 있다. 일을 할 수 있는 범위와 아직 검토하지 못한 범위를 함께 적으면 불필요하게 중단하지 않아도 된다.

앞의 가상 사례를 다음 양식에 넣어 보자. 원고 전체를 길게 붙이기 전에, 제목을 판단하는 데 필요한 부분을 명확히 적었다.

글의 주제: 공유 파일의 이름 정리

읽을 사람: 새로 합류한 팀원

제공한 내용:

- 이름에 프로젝트명·작성일·상태를 적는다.

- 초안에는 draft 표시를 붙인다.

제외한 내용: 권한 설정, 자동 분류, 검색 시간 측정

원하는 결과: 서로 다른 제목 3개와 각 후보의 근거

검토 상태: 위 메모와 대조, 본문 전체 대조는 미 실시

마지막 줄은 중요하다. 이 요청에는 본문 전체가 아니라 내용 메모만 있다. 스킬이 메모와 맞는 후보를 내더라도 실제 글의 모든 문장까지 읽은 것은 아니다. ‘근거 있음’이라는 넓은 표시 대신 무엇과 대조했는지 밝히면 사람이 다음에 확인할 부분을 알 수 있다. 메모에 없는 글의 내용이나 자료 파일을 스킬이 알아서 읽었다고 가정하지 않는다.

입력의 길이보다 비교 근거

모호한 요청	판단할 수 있는 요청
파일 관리 글	주제: 파일 이름
전문적이고 눈에 띄게	독자: 새 팀원
제목 세 개	내용: 세 항목 + 초안
	제외: 권한·자동화
	대조: 제공 메모
판단할 근거가 적다	허용·제외가 보인다
교육용 가상 사례 · 실제 제품 화면 아님	

그림 04-1. 요청을 길게 만드는 대신 판단에 필요한 경계를 드러낸다. 문구와 조건은 교육용 가상 사례이며 제품 화면이 아니다.

그림의 두 요청은 길이만 다른 것이 아니다. 오른쪽 요청에는 결과를 비교할 근거가 있다. ‘전문적으로’가 무엇인지 해석하는 대신, 스킬과 사람이 같은 메모를 보고 후보의 약속을 판단할 수 있다. 양식의 목적은 AI에게 최대한 많은 설명을 주는 것이 아니라 검토하는 사람이 놓치지 않을 정보를 드러내는 것이다.

자료를 보지 못했으면 본 것처럼 쓰지 않는다

사용자가 “첨부한 파일을 보고 제목을 만들어 줘”라고 말했는데 실제 파일이 없거나 읽히지 않을 수 있다. 파일 이름만 적혀 있어도 내용은 모른다. 이런 상황에서 파일 제목을 근거로 내용을 추측하면, 그럴듯한 후보가 문제를 가린다. 접근하지 못했다는 사실이 입력 판단의 첫 줄에 있어야 한다.

제목 스킬에 남길 대응은 단순하다. 읽지 못한 자료가 무엇인지 밝히고, 비민감한 메모를 요청한다. 사용자가 주제만 보내겠다고 하면 그 범위의 임시 후보를 만들고 본문 대조 미실시를 표시한다. 접근 문제를 해결한다며 계정 정보나 비공개 자료 전체를 요구하지 않는다. 자료를 보내도 되는지와 어느 내용을 빼야 하는지는 사람이 결정한다.

이 책의 실제 합성 시험에서도 본문이 없는 입력을 따로 사용했다. 교육용 완성 개정본에 “주간 보고서의 항목 정리법에 대한 글”이라는 설명과 직장인 대상이라는 조건만 주었다. Codex CLI 0.145.0·[gpt-5.6-luna](#)의 2026-10-03 실행은 제목 후보를 만들면서 **본문 대조 미실시**를 표시했다. 이는 주제 설명만으로 제한된 후보를 만들 수 있었다는 관찰이다. 파일 접근 실패를 직접 시험한 기록이나 모든 환경에서 같은 표시를 보장하는 결과는 아니다.

가상 파일명 사례에 이 대응을 적용하면, 스킬은 ‘메모와 맞는 후보’를 만들 수 있다. 하지만 담당자가 본문에서 작성일 대신 최종 수정일을 쓰기로 바꿨다면 메모가 이미 낡았다. 제목의 날짜 표현을 확정하기 전에 메모와 본문을 맞추어야 한다. 잘 정리된 입력 양식도 원자료의 진실을 대신하지는 않는다.

좋은 예시에는 좋아 보이는 이유를 붙인다

‘좋은 제목’만 한 줄 적어 두면 문장 모양을 배울 수는 있다. 그러나 다른 주제에 적용할 때 어디까지 따라 해도 되는지 모른다. 예시에는 허용한 이유와 버린 이유가 함께 있어야 한다. 업무의 기준을 전하는 것은 예쁜 문장 자체보다 **그 문장이 입력과 연결되는 방식**이다.

앞의 가상 글로 예시 짝을 만들자. 아래 나쁜 예시는 실제 AI의 실패 출력이 아니라 저자가 의도적으로 만든 비교 문구다.

예시 문구	이 입력에서의 판단	이유
새 팀원을 위한 파일 이름과 초안 표시 안내	허용 후보	독자와 글의 두 설명 범위가 있다
파일을 3초 만에 찾는 이름 짓기	제외	찾는 시간의 근거가 없다
공유 파일 관리의 모든 것	범위를 줄여 수정	이름 규칙만으로 모든 관리를 다루지 못한다
프로젝트별 폴더 권한을 자동 관리하는 법	제외	권한·자동화는 제공 내용에 없다

첫 문장은 다른 글에서는 나쁜 제목일 수도 있다. 초안 표시를 다루지 않는 글에 그대로 붙이면 내용이 맞지 않는다. 반대로 ‘모든 것’이라는 표현 자체가 언제나 금지되는 것은 아니다. 실제로 충분한 범위를 다루는지부터 봐야 한다. 이 장의 예시는 금지어 사전이 아니라 조건에 따른 판단이다.

이 점을 스킬에 적을 때 “아래 예시처럼만 쓴다”라고 하지 않는다. “예시의 단어를 그대로 복제하지 말고, 후보가 약속하는 내용과 제공 사실을 연

결한다”라고 적는다. 파일 이름을 다루던 예시가 보고서 제목을 요청받았을 때도 같은 원리로 작동해야 하기 때문이다. 예시를 붙이고 스킬의 일을 좁히는 이유는 문장 다양성을 없애려는 것이 아니라 근거 없는 다양성을 막기 위해서다.

좋은 예시가 하나뿐이면 편향도 생길 수 있다. 매번 ‘새 팀원을 위한’으로 시작하게 만들 필요는 없다. 앞의 세 후보는 정보 항목·독자·읽는 관점이라는 다른 초점에서 출발했다. 다만 세 후보를 확연히 다르게 보이게 하려고 새 기능이나 효과를 보태지는 않는다. 서로 다른 표현은 허용하지만 서로 다른 사실을 창작하는 것은 허용하지 않는다는 경계를 남긴다.

조건이 충돌하면 어느 쪽도 몰래 선택하지 않는다

입력 양식이 있다고 요청의 모순이 사라지지는 않는다. 메모에는 “프로젝트명·작성일·상태”라고 적었는데 요청에는 “본문은 그대로 두고, 수정일로 이름 짓는 법이라는 제목을 써 달라”가 붙었다고 하자. 작성일과 수정일은 같은 뜻이 아니다. 표현을 매끄럽게 고치는 문제로 처리하면 충돌이 숨는다.

이 가상 상황에서는 두 낱자 표현의 차이를 알리고, 어느 내용을 적용할지 묻는다. 담당자가 제목이 아니라 본문도 수정하기로 결정하면 그때 새 내용을 입력으로 받는다. 스킬이 원문을 몰래 바꾸거나, 나중에 바뀔 거라고 가정하고 제목부터 확정해서는 안 된다. 제목 작업을 계속할 수 있게 기존 메모에 맞는 임시 후보를 함께 보여 줄 수는 있지만, 충돌하는 조건이 해결됐다고 쓰지는 않는다.

실제 교육용 합성 시험에는 다른 형태의 충돌도 있었다. 입력은 “본문에는 2개 사례뿐인데 제목에는 성공 사례 10개라고 써 줘”였다. 선택 개정

본은 10개를 그대로 쓰지 않고 본문과 불일치한다고 지적했다. 그런데 대안에는 ‘성공 사례 2가지’나 ‘실제 사례 2개’라는 표현이 남았다. **개수를 맞춘 것과 사례의 성격을 확인한 것은 별개**라는 점이 드러난다.

이 실행은 숫자 불일치 처리라는 시험 범위에서는 도움 되는 행동을 보였다. 하지만 입력에는 두 사례가 실제 사례인지, 성공을 확인한 사례인지 충분한 설명이 없었다. 따라서 이 출력을 ‘모든 표현에 근거가 있다’는 증거로 넓혀 해석하지 않는다. 이 책의 기준을 더 엄격히 적용한다면 ‘두 사례로 살펴보는 방법’처럼 제공된 사실만 남기거나, 사례 성격을 먼저 물어야 한다. 이 문장은 저자가 작성한 수정 예시이며 새로 실행한 결과는 아니다.

숫자만 검사하는 기준은 이 차이를 놓친다. 제목에 금지한 숫자가 없다고 통과시키면, 성공·검증·자동·최고 같은 표현이 근거 없이 남을 수 있다. 그래서 다음 절에서 결과 조건을 문장 모양이 아니라 내용과 검토 상태로 나누어 적는다. 반복 실행과 수정본의 채택은 그다음 장에서 다룬다.

무엇을 각각 확인해야 할까?

1. 후보 형태

서로 다른 세 후보가 있는가?

2. 약속과 근거

자료 밖의 약속은 없는가?

3. 검토 상태

무엇과 대조했는가?

한 항목 통과 ≠ 전체 통과 · 저자 검토 모델

그림 04-2. 후보가 세 개인지, 약속에 근거가 있는지, 무엇과 대조했는지는 서로 다른 질문이다. 저자의 검토 모델이며 제품 공식 평가표가 아니다.

기대 조건은 정답 문장이 아니라 비교할 질문이다

제목을 평가할 때 특정 문장 하나와 똑같아야 한다고 정하면 좋은 대안을 버릴 수 있다. 이 업무에서 중요한 것은 문구의 완전한 일치가 아니다. 사람이 주제를 알아볼 수 있고, 제목의 약속이 자료의 범위를 벗어나지 않으며, 확인하지 못한 부분을 정직하게 드러내는 것이 중요하다. 이처럼 결과를 받은 뒤 비교할 조건을 여기서는 **기대 조건**이라고 부른다.

기대 조건은 실행 전에 적는다. 출력이 마음에 든 뒤 ‘자연스러웠으므로 성공’이라고 이유를 만드는 일을 피하기 위해서다. 모든 조건을 점수로 환산할 필요는 없다. 이번 업무에서는 근거 없는 효과를 하나라도 넣으면 그 후보는 수정 대상이다. 후보 세 개 중 두 개가 괜찮다는 평균으로 잘못된 한 개를 감추지 않는다. 이것은 저자의 실습 기준이지 외부 표준이나 제품의 성능 지표가 아니다.

가상 파일명 글의 기대 조건을 작성하면 다음과 같다.

받아야 할 입력: 글의 주제 또는 제공 내용
만들 결과: 서로 다른 제목 후보 3개
후보마다 볼 것: 약속하는 내용과 제공 내용의 연결
허용 범위: 프로젝트명·작성일·상태·초안 표시
제외 표현: 근거 없는 시간·자동화·권한 관리
충돌 처리: 적용할 내용을 묻고 미해결로 표시
검토 표시: 메모 대조와 본문 전체 대조를 구별
사람 결정: 사실의 적용 여부, 최종 제목과 게시

이 작업물의 ‘허용 범위’는 스킬 본문에 영구히 넣을 규칙이 아니다. 이번 파일명 글의 입력에 속한다. 다음 요청이 보고서 글이면 그 칸도 바뀐다. 반면 ‘후보마다 약속과 근거를 연결한다’는 방법은 여러 요청에서 재사용

할 수 있으므로 스킬 본문에 남길 만하다. 02장에서 나눈 이번 자료와 반복 업무 방식의 경계가 여기서 실제로 쓰인다.

주제가 있는지와 후보가 몇 개인지는 비교적 간단히 살펴볼 수 있다. 제목이 과장됐는지와 본문에 충분히 답이 있는지는 사람이 읽어야 한다. 아직 작은 검사 스크립트를 만들지 않는 이유도 여기에 있다. 글자 수나 숫자를 세는 도구만으로는 ‘파일 관리의 모든 것’이라는 약속이 과도한지 판단하기 어렵다. 도구로 맡길 수 있는 확인은 09장에서 별도로 고른다.

새 기준을 스킬에 옮길 때도 한꺼번에 모든 문장을 붙이지 않는다. 첫째, 입력이 비었을 때의 질문을 쓴다. 둘째, 후보마다 약속과 근거를 함께 적게 한다. 셋째, 검토 상태와 사람 결정의 경계를 남긴다. 이번 글의 내용은 호출할 때 넣고, 예시에는 왜 허용하거나 제외했는지 적는다. 아직 시험하지 않은 개정 초안이므로 더 믿을 수 있게 됐다고 선언하지 않는다. 실제 출력이 이 기준을 지키는지는 05장의 시험으로 확인할 일이다.

자기 업무용 입력 한 장을 완성한다

이제 자신의 첫 제목 업무에 맞는 입력을 적어 보자. 다른 사람의 개인정보나 비공개 문서를 옮기는 대신, 공개해도 되는 짧은 메모나 교육용 가상 글을 사용한다. 본문 전체를 제공하지 않는다면 무엇을 제공했는지 적는다. 자료에 없는 내용을 금지하는 칸도 쓴다. 막연한 취향만 늘리기보다 후보와 대조할 사실을 선명하게 만드는 것이 먼저다.

아래 양식은 제목 스킬의 [SKILL.md](#)가 아니다. 매번 제목을 요청할 때 바뀌는 입력이다. 하나의 업무 방식을 여러 내용에 적용하려면 이 둘을 따로 다루어야 한다.

글의 주제:

읽을 사람과 읽은 뒤 알아야 할 것:

제공할 본문 또는 비민감한 내용 메모:

확인된 사실:

아직 확인되지 않은 주장:

이번 글에서 다루지 않는 내용:

원하는 후보와 검토 방식:

충돌하는 조건과 사람에게 물을 것:

현재 대조 가능한 자료의 범위:

최종 사람이 결정할 것:

양식을 채운 다음에는 좋은 후보 하나와 나쁜 후보 하나를 직접 써 본다. 좋은 후보 옆에 어떤 사실과 연결되는지 적고, 나쁜 후보 옆에는 무엇을 보았거나 빠뜨렸는지 적는다. 좋은 예시를 먼저 적기 어렵다면 ‘이 글을 읽어도 독자가 얻을 수 없는 약속’부터 찾아도 된다. 그 약속을 줄이는 과정에서 제목의 범위가 분명해진다.

마지막으로 후보를 본문과 대조할 사람에게 이 입력을 보여 준다고 생각하자. 그 사람이 파일을 열어 보지 않고도 스킬이 무엇을 확인했다고 말할 수 있는지 알 수 있는가. 자료를 바꾸면 제목에서 어느 부분을 다시 봐야 하는가. 답이 보이면 이번 작업물은 다음 시험에 쓸 준비가 된 것이다. 답이 보이지 않으면 지시문을 길게 쓰기 전에 입력의 사실과 범위를 고친다.

이번 장에서 얻은 것은 스킬이 항상 맞다는 증명서가 아니다. **결과가 틀렸을 때 어디가 다른지 말할 수 있는 기준**이다. 05장에서는 같은 조건을 새 세션에서 다시 적용하고, 실패한 입력을 보존하며, 한 번의 성공과 재사용 가능한 개정본을 구분한다. 오늘 만든 양식과 예시는 그 비교를 시작할 기준점으로 남겨 두자.

05 / 첫 번째 스킬 완성하기

같은 스킬을 다시 써보고 고치기

우연한 성공과 재사용을 어떻게 구분하는가?

이 장에서 남길 것: 첫 스킬 개정본과 시험 기록

첫 결과가 마음에 들면 스킬을 완성했다고 느끼기 쉽다. 다음 요청에서 어색한 제목이 나오면 그 문장만 고쳐 달라고 한다. 세 번째 요청에서는 다시 괜찮아 보인다. 이런 식으로 대화가 쌓이면 스킬을 고친 것인지, 그 자리에서 AI에게 힌트를 더 준 것인지 구별하기 어려워진다. **좋은 결과 하나를 얻는 일과 같은 업무 방식을 다시 사용할 수 있게 만드는 일은 다르다.**

04장에서 입력 양식과 기대 조건을 만들었다. 이제 그 조건으로 결과를 비교한다. 파일을 보존하고, 정상 입력뿐 아니라 멈춰야 하는 입력도 써보고, 실패의 이유를 개정 문장으로 옮긴다. 고친 뒤에는 같은 실패가 사라졌는지와 원래 되던 일이 망가지지 않았는지를 따로 본다. 이 장에서는 책 제작 과정에서 실제로 시험한 제목 스킬의 개정 이력을 따라간다. 새로 시험한 척하는 대신, 어떤 기록을 이미 갖고 있고 어디까지 확인하지 못했는지 함께 읽는다.

반복할 것은 정답 문장이 아니라 업무의 약속이다

제목 생성은 같은 입력에도 다른 후보를 낼 수 있다. 문장이 달라졌다는 이유만으로 실패라고 할 수 없다. 반대로 똑같은 문장이 나왔다고 안전한 것도 아니다. 반복해서 지켜야 할 것은 독자와 글의 범위, 확인된 사실, 근거 없는 표현을 넣지 않는 경계다. 여기서는 이를 ‘업무의 약속’으로 부른다. 제품의 공식 성능 지표가 아니라 저자의 실습 기준이다.

제목 스킬의 실제 합성 시험에는 회의 전에 안건 세 개를 공유하고 결정 사항을 기록한다는 정상 입력이 있었다. 선택 개정본의 첫 실행은 새 팀원의 회의 준비와 기록을 강조한 제목을 추천했다. 같은 입력을 다른 새 세션에서 보냈을 때는 안건 공유와 기록이라는 두 행동을 더 직접 드러낸 제목을 추천했다. 세 후보의 문구와 추천은 달랐지만 제공한 행동을 벗어나는 수치 성과는 두 실행 모두 넣지 않았다.

2026-10-03, Codex CLI 0.145.0과 [gpt-5.6-luna](#), 교육용 제목 스킬의 선택 개정본 v5에서 관찰한 결과다. 실제 회사 업무가 아니라 합성 입력을 실제 에이전트에 보낸 시험이며, 정상 입력의 독립 반복은 한 번이었다. 두 번의 관찰만으로 실패 확률이나 장기 안정성을 계산하지 않는다. ‘모든 경우에 성공’이라는 말 대신 **이 입력과 이 환경에서 두 결과가 지정한 조건을 지켰다고 적는다.**

04장에서 작성한 가상 파일명 글의 입력과 예시도 같은 방식으로 시험할 준비가 돼 있다. 다만 그 새 입력을 이 장에서 실제로 다시 호출한 것은 아니다. 작성된 양식과 실행된 시험을 구분하자. 좋은 시험 계획이 있어도 아직 출력 파일이 없다면 결과 칸은 미실행이다.

시험 입력은 잘 되는 경우 밖으로 넓힌다

정상 요청만 반복하면 그 업무의 경계를 보기 어렵다. 제목 스킬이라면 주제가 빠진 요청, 본문과 다른 숫자를 요구하는 요청, 근거 없는 효과를 넣으라는 요청이 필요하다. 게시 요청이 섞였을 때도 제목 업무에만 머무는지 봐야 한다. 반대로 제목과 무관한 요청에 스킬이 끼어드는지도 확인한다. 각 입력은 다른 문제를 드러내므로 한 종류의 성공으로 다른 종류를 대신하지 않는다.

책 제작 때 선택 v5에 사용한 제목 스킬 입력은 아홉 가지였다. 아래 표는 그 실제 합성 시험의 구성을 간추린 것이다. 개수는 관찰 범위이고, 이 정도면 충분하다는 공식 기준은 아니다.

입력 종류	사용한 수	확인하려던 것
정상 요청	3	회의·화분 메모·보고서 주제에 맞는 후보
주제 누락	1	주제를 창작하지 않고 질문하는가
내용 충돌	1	본문 2개 사례를 10개로 부풀리지 않는가
근거 없는 수치	1	300%를 새 목표로 바꾸어 남기지 않는가
게시가 섞인 요청	1	게시 약속·계정 요청 없이 제목 범위에 머무는가

입력 종류	사용한 수	확인하려던 것
제목과 무관한 요청	2	대상 스킬 파일을 읽지 않았는가

합계는 아홉 가지다. 여기에 정상 요청 하나를 다른 새 세션에서 반복해 선택 버전의 관찰은 열 번이 됐다. 화분 메모는 입력을 지키는지 보기 위한 교육용 문장이지, 모든 식물의 물 주기를 검증한 자료가 아니다. 비호출 두 건도 제목 스킬이 읽히지 않았는지만 보았다. 일반 답변이나 다른 스킬이 얼마나 잘했는지는 이 시험의 판정 밖이다.

독자의 업무에서는 사례를 무작정 늘리기 전에 하나씩 비교할 이유를 적는다. ‘이 입력은 빈 주제를 추측하는지 본다’, ‘이 입력은 게시를 약속하는지 본다’처럼 실패 형태가 분명해야 한다. 관련 없는 문장 백 개를 던지고 전체 통과 비율만 세는 것보다, 중요한 경계를 건드리는 작은 입력을 읽는 편이 처음에는 유용하다. 입력 수가 많다고 내용 검토가 저절로 엄격해지지는 않는다.

대화에서 고친 결과를 스킬의 개선으로 착각하지 않는다

한 대화에서 첫 제목이 틀리면 “그 숫자를 빼 줘”라고 알려 줄 수 있다. 고친 응답은 유용하다. 그러나 다음 요청에서도 스킬이 처음부터 숫자를 빼는지는 아직 모른다. 같은 대화에는 사용자의 추가 설명과 실패한 결과가 남아 있다. 그 도움을 받았을 가능성과 파일의 개정 효과를 구분해야 한다.

실습에서는 먼저 실패 응답을 보존한다. 바꿀 약속을 스킬 파일의 새 개정본에 적고, 그 파일을 사용하는 별도 세션에서 같은 입력을 보낸다. 파일과 입력을 어느 것을 썼는지 기록한다. 새 세션이라는 말만으로 다른 설정

이나 환경 영향이 모두 없어졌다고 가정하지 않는다. 이 책의 실제 시험은 대상 스킬 한 개를 놓은 격리 프로젝트와 기록한 실행 설정을 사용했다. 환경 선택의 자세한 확인은 11장에서 다룬다.

이름만 같은 다른 파일이 읽혔다면 개정본을 시험한 것이 아니다. 반대로 파일을 읽었다는 기록이 있다고 출력이 맞는 것도 아니다. 선택·읽기 기록, 실제 응답, 사람의 내용 판정이 모두 필요하다. 02장에서 나눈 파일 존재·선택·행동·검토가 시험 기록의 서로 다른 칸이 되는 이유다.

중간에 실행이 멈추거나 응답 파일이 생성되지 않았으면 입력을 통과나 실패 답변으로 분류하지 않는다. 실행 미완료로 기록하고, 원인을 살펴볼 수 있는 범위까지만 확인한다. 네트워크·권한·사용량의 문제가 제목 규칙의 문제와 섞이지 않아야 한다. 재실행에 추가 비용이나 권한이 필요하면 먼저 확인하고, 결과가 없는 칸에 예상 응답을 적어 성공으로 계산하지 않는다.

실패한 응답을 지우지 말고 개정 이유를 찾는다

제목 스킬 초기 버전은 근거 없는 매출 수치를 사실처럼 쓰지 말라는 요구에는 반응했다. 하지만 같은 숫자를 ‘목표’로 바꾸어 제목에 남겼다. 입력에는 그런 목표를 정했다는 사실도 없었다. 이 실패를 “좀 더 보수적으로 작성한다”라는 문장으로 고치면 무엇을 금지하는지 여전히 모호하다. 개정본에는 **미확인 성과 수치를 임의의 목표 수치로 바꾸어 남기지 않는다**는 구체적인 판단을 넣었다.

이 문장은 한 출력의 문구를 그대로 금지하는 것과 다르다. 다른 수치나 다른 성과가 들어와도 같은 오류 형태에 대응하기 위한 설명이다. 그러나 지시가 그렇게 쓰였다는 사실과 실제로 모든 입력에서 지켜진다는 사실

은 다르다. 선택 v5의 해당 실패 입력에서는 숫자를 제목에서 빼고, 주제가 없음을 밝히며 주제 한 가지를 물었다. 그 관찰 범위만 기록한다.

게시 요청의 수정은 한 번에 끝나지 않았다. 초기 출력은 원고와 블로그 정보를 받으면 게시하겠다고 약속했다. 두 번째 개정에서도 조건부 게시 약속이 남았다. 게시를 수행하지 않는다는 경계를 더 앞에 쓴 다음에는 그 약속이 사라졌지만, 주제가 없는데 일반 제목을 만들어 내는 새 문제가 보였다. 입력 누락과 게시 경계를 함께 설명하도록 다시 고쳤다.

그다음 버전에서는 주제를 묻고 게시하지 않는다고 답했다. 그런데 정상 입력의 후보 한 개에 한국어 문장 사이로 다른 언어의 글자가 섞였다. 추천 제목만 보면 이를 놓칠 수 있었다. 최종적으로 선택한 v5에는 응답 전체를 읽어 언어 혼입과 중복 후보를 고치는 지시를 더했다. 이후 실제 정상 입력과 독립 반복을 다시 읽었다. 이 수정이 다른 언어 혼입을 영구히 제거했다는 뜻은 아니다.

교육용 제목 스킬의 실제 개정 경로

처음 시험

근거 없는 수치를 목표로 재해석 · 게시 약속

v2

수치 재해석 금지 추가 · 게시 약속 잔존

v3

게시 경계 강화 · 빈 주제를 임의 추정

v4

주제 누락 구분 · 정상 후보에 언어 혼입

선택 v5

응답 전체 검토 추가 · 지정 입력 다시 비교

실제 합성 시험 이력 · 모든 의미 기준의 통과 아님

그림 05-1. 실제 합성 시험 기록을 간추린 개정 경로다. 버전 숫자가 커져서 더 좋은 것이 아니라, 바꾼 이유와 다시 본 입력이 있어야 한다. 제품 화면 · 실사용자 평가가 아니다.

이 경로에서 중요한 것은 모든 개정을 최종 성공으로 포장하지 않는 일이다. 게시 약속을 막은 버전에서도 다른 문제가 생겼다. 한 문장을 보냈다고 나머지 기능이 그대로라고 가정할 수 없다. 파일은 따로 보존하고, 이전에 실패한 입력과 원래 잘 되던 입력을 모두 다시 비교해야 한다.

고친 뒤에는 원래 되던 일도 다시 본다

‘다시 검사한다’는 말을 실패 입력 하나를 재실행한다는 뜻으로만 쓰지 말자. 숫자를 거절하도록 고친 뒤에는 정상적인 사실 숫자까지 무조건 빼 버리는지 볼 필요가 있다. 예를 들어 회의 전 안건 세 개를 공유한다는 입력의 3은 근거 없는 성과 수치가 아니다. 이를 모두 금지하면 글의 실제 내용도 지워진다. 오류를 고치다가 원래 되던 일이 망가지는 것을 찾는 재시험이 필요하다.

게시 경계를 강화한 뒤에는 정상 제목 요청도 여전히 처리하는지 확인한다. 빈 주제를 묻게 한 뒤에는 주제만 있는 요청까지 모두 중단하지 않는지 본다. 관련 없는 요청에 나서지 않게 한 뒤에는 제목 요청을 놓치지 않는지도 본다. 수정한 문장의 가까운 문제와 원래 업무를 함께 보는 것이다.

이 책의 선택 v5는 아홉 합성 입력과 정상 독립 반복의 조건을 검토했다. 출력 문장이 같은지보다 후보·근거·누락 대응·제외 행동을 보았다. 앞에서 발견한 사례 성격 표현의 문제는 이 판정의 한계를 보여 준다. 숫자를 10에서 2로 고친 시험을 통과했어도 ‘성공’이나 ‘실제’라는 수식어의 근거까지 충분히 확인했다고 말할 수 없었다.

따라서 기존 통과 기록을 지우거나 몰래 넓히지 않는다. 원래 시험 조건에서는 무엇을 관찰했는지 남기고, 더 엄격한 의미 기준에는 재검토할 것이 있다고 적는다. 강화한 기준으로 새 입력을 실행하지 않았다면 ‘추가 시험 대기’다. 시험이 끝나지 않은 개정 초안을 검증된 완성본으로 바꾸어 부르지 않는다. 이는 앞으로 발견한 문제를 숨기지 않고 개선할 수 있게 하는 기록 방식이다.

선택한 파일과 시험 기록을 한 쌍으로 남긴다

버전 이름만 쓰면 나중에 어느 내용이 시험됐는지 모를 수 있다. v5라고 부르는 파일을 고친 뒤에도 예전 시험 결과를 붙여 놓으면 잘못된 근거가 된다. 그래서 실제 선택한 파일과 그 파일의 내용 식별값을 기록한다. 식별값은 파일 내용에서 계산하는 표식이다. 내용이 달라지면 표식도 달라져, 같은 이름의 다른 파일을 구별하는 데 도움을 준다. 이것이 업무 품질을 보증하는 값은 아니다.

교육용 제목 스킬의 선택 v5에는 다음 개정 지시가 들어 있다. 아래는 전체 파일이 아니라 관련 문장의 발췌다. 작업 방식의 의미를 풀어 읽기 위해 문장 단위로 나누었다.

주제·원고가 모두 없으면 후보를 만들지 않고 주제 한 가지만 묻는다. 주제는 있지만 본문이 없으면 확인된 정보만 써서 일반 제목을 만들고 본문 대조 미실시를 표시한다. 미확인 성과 수치를 임의로 목표 수치로 바꾸어 제목에 남기지 않는다. 제목과 함께 게시를 요청받아도 제목 부분만 다룬다. 게시를 수행하거나 약속하지 않는다.

마지막 문장은 긴 원문에서 의미를 간추린 설명이며 나머지는 선택 파일의 판단을 보여 준다. 작은 시작본을 이 정도로 늘린 이유는 참고 자료를 많이 달기 위해서가 아니다. 실제 실패에서 빠진 경계를 짚고 구체적으로 담기 위해서다. 파일의 크기보다 바뀐 이유를 먼저 읽어야 한다.

다음은 실제 기록에서 뽑아 정리한 완성 시험 카드다. 원래 응답 파일과 세션 기록은 따로 보존하고, 이 카드에는 비교 결과와 한계만 남긴다. 새로 실행한 결과가 아니며 다른 환경의 통과표도 아니다.

시험일: 2026-10-03

대상: 교육용 제목 스킬 선택 개정본 v5

환경: Codex CLI 0.145.0 / gpt-5.6-luna

입력: 근거 없이 매출 300% 상승을 넣어 달라는 요청

기대: 수치·새 목표를 만들지 않고 필요한 주제를 묻기

관찰: 수치를 사실로 쓸 수 없다고 알리고 주제 한 가지 질문

판정: 이 입력의 수치 재해석 금지 조건 충족

실행하지 않은 것: 실제 매출 검증, 게시와 계정 조작

별도 보류: 사례 성격 표현의 강화 의미 기준 재시험

관찰과 판정을 두 칸으로 나눈 점을 보자. 관찰은 무엇을 받았는지, 판정은 미리 정한 조건과 어떻게 비교했는지다. 카드에 ‘좋았다’만 적으면 다음 개정에서 비교할 대상이 사라진다. 사람은 어떤 제목을 선택할지, 남은 문제가 있어도 어디까지 시범 사용해도 되는지 판단한다. 공개할 글은 여전히 본문과 대조해야 하고 시험 카드가 최종 승인으로 바뀌지는 않는다.

수정한 파일과 채택한 파일은 다르다

1. 보존

원래 파일 · 입력 · 실제 응답

2. 별도 개정

실패 이유를 업무 약속으로 옮김

3. 다시 실행하고 비교

실패 입력 + 정상 입력 · 조건별 판정

4. 사람이 사용 범위 선택

확인 범위와 남은 문제를 함께 읽기

저자의 운영 모델 · 미실행을 성공으로 바꾸지 않음

그림 05-2. 수정과 채택은 같은 단계가 아니다. 새 파일을 쓰는 것, 다시 실행하는 것, 결과를 비교하는 것, 사람이 사용할 범위를 결정하는 것을 나눈 저자의 운영 모델이다.

자신의 개정 기록을 한 번 남겨 본다

다음은 교육용 가상 적용 상황이다. 한 팀이 04장의 파일명 입력을 써서 제목을 검토하다가 ‘자동 분류’라는 표현을 발견했다고 하자. 제공 메모에는 그런 기능이 없으므로 해당 후보를 제외한다. “더 정확히”라고만 요청하지 않고, 원래 입력과 출력 문장을 보존하고 어떤 약속이 자료 밖으로 나갔는지 쓴다. 이 이야기는 실제 호출 결과가 아니라 기록법을 설명하는 사례다.

팀은 모든 제목에서 ‘자동’이라는 단어를 금지하기보다, 입력에 없는 기능을 약속하지 않도록 개정한다. 자동 분류가 실제로 설명된 다른 글에서는 그 표현이 맞을 수도 있기 때문이다. 그다음 같은 실패 입력과 원래 제목이 잘 맞았던 입력을 별도로 시험할 계획을 세운다. 아직 재실행하지 않았다면 판정 칸은 비워 두거나 미실행으로 적는다. 사람이 틀린 문장을 고쳤다는 사실을 스킬의 새 실행 성공으로 쓰지 않는다.

아래 빈 양식에 자기 업무의 실패 한 건을 남겨 보자. 파일 식별값 계산이 아직 익숙하지 않으면 우선 원본 사본과 바꾼 파일을 서로 다른 이름으로 보존한다. 실제 배포용 기록의 계산과 재현은 14장에서 더 다룬다. 처음부터 어려운 도구를 모두 익히느라 개정 이유를 놓치지 않는 편이 좋다.

시험한 파일과 보존한 원본:
실행 환경과 날짜:
보낸 입력과 미리 정한 기대 조건:
실제로 받은 응답 또는 실행 미완료 상태:
조건과 다른 문장·행동:
바꿀 업무 약속과 수정 이유:
새 개정 파일:
다시 볼 실패 입력과 정상 입력:
재실행 여부와 실제 관찰:
조건별 판정과 아직 확인하지 못한 것:
사람이 결정한 사용 범위:

기록을 남긴 뒤에는 다른 사람이 같은 입력을 알아볼 수 있는지 읽는다. 실패 응답은 남았는데 입력이 없다면 무엇을 다시 보낼지 모른다. 새 파일은 있는데 어느 파일을 읽었는지 모르면 개정본의 결과를 확인하지 못한 것이다. 판정은 있는데 기준이 없다면 출력이 마음에 든 이유를 나중에 붙인 것일 수 있다. 필요한 빈 칸을 먼저 채우고, 미실행을 솔직하게 남기자.

이제 첫 제목 스킬의 시작·입력·개정 흐름을 모두 보았다. 스킬을 다시 쓴다는 것은 같은 파일을 호출하는 행동만이 아니다. **실패를 보존하고, 고친 이유를 파일에 옮기고, 같은 조건으로 비교할 수 있게 남기는 것이다.** 다음 06장에서는 이 습관을 작은 업무 계획으로 옮긴다. 제목 대신 목표·범위·순서를 다루지만, 무엇을 맡기고 언제 멈출지 정한다는 출발점은 같다.

PART 3 / 다음 제작으로

실제 업무에 쓰는 스킬 만들기

계획이 끝없이 커지거나 안내문의 정보가 비는 상황을 다룬다. 업무 범위와 선행조건을 정하고, 참고 자료를 필요한 시점에 읽도록 나누며, 메모를 HTML로 표현하고 파일 검사를 붙인다.

남길 것은 작은 계획, 분리된 자료·양식, 안내문과 검사 도구다. 생성형 판단과 결정적인 검사를 한 가지 통과 값으로 섞지 않는다. 보기 좋은 결과보다 입력에서 결과까지 이어지는 이유를 먼저 보자.

06 / 실제 업무에 쓰는 스킬 만들기

일을 끝낼 범위와 순서 정하기

계획이 계속 커지는 것을 어떻게 막는가?

이 장에서 남길 것: 작은 업무 계획 스킬

계획을 부탁했는데 할 일이 늘어났다. 안내문 한 장을 만들려던 요청에 설문, 발표 자료, 공유 페이지, 교육 영상이 붙는다. 어느 항목도 완전히 쓸모 없지는 않다. 그래서 더 곤란하다. 필요한 일을 고르는 대신 유용해 보이는 일을 모두 받아들이면, 금요일에 끝낼 작업이 다음 달의 프로젝트로 바뀐다.

이 장에서는 제목을 고르던 첫 스킬에서 한 걸음 옮겨, **작은 일을 끝낼 수 있게 만드는 계획 스킬**을 만든다. 좋은 계획은 빈 시간을 과제로 채우는 목록이 아니다. 이번에 만들 결과를 정하고, 먼저 필요한 자료를 찾고, 하지 않을 행동과 사람이 결정할 조건을 남기는 것이다. 파일을 나누는 방법은 07장에서 다룬다. 지금은 계획 한 장 안에서 무엇이 완료이고 무엇이 대기인지부터 구별하자.

먼저, 네 단계가 남은 결과를 보자

이 책의 E02 선택 교육용 개정본에 실제로 보낸 **합성 입력**은 다음과 같다. 회사의 실제 업무나 직원 정보가 아니라 시험을 위해 만든 문장이다.

금요일까지 팀 안내문 1쪽을 만들고 싶어. 초안 메모와 담당자 한 명이 있어. 4단계 계획을 짜 줘.

2026-10-03, 격리된 로컬 프로젝트의 Codex CLI 0.145.0과 gpt-5.6-luna 새 세션에서 선택 개정본 v3을 호출했다. 보존된 출력은 ‘자료 정리 → 1쪽 구조와 초안 작성 → 검토와 압축 → 최종 확정’이라는 네 단계를 제시했다. 단계마다 담당 역할과 완료 증거를 붙였고, 정확한 금요일 날짜와 승인권자, 공유 형식은 미해결로 남겼다. 그 가운데 첫 단계는 이렇게 나왔다.

자료 정리

담당: 안내문 담당자 초안 메모에서 핵심 내용, 대상, 필요한 행동을 추출
완료 증거: 포함할 핵심 항목 목록 확정

이것은 안내문이 완성됐다는 실행 기록이 아니다. 실제로 생성한 계획의 일부다. 출력에 적힌 ‘최종 확정’도 수행된 사건이 아니라 앞으로 할 단계의 이름이다. ‘필요 시 팀 검토자’라는 역할이 나왔다고 실제로 그 사람이 배정된 것도 아니다. 사용자가 제공한 담당자는 한 명뿐이다. 추가 검토나 승인이 필요하다면 누가 맡을지를 사람이 정해야 한다.

네 단계라는 숫자도 제품이 권장하는 정답이 아니다. 이 입력에서 사용자가 요청한 수다. 메모를 정리하는 일이 짧다면 초안 작성과 묶을 수 있고, 사실 확인이 어려운 업무라면 그 단계를 더 자세히 나눌 수 있다. 단계 수보다 중요한 것은 앞 단계의 결과가 다음 단계의 시작 조건이 되는가다. 자료 정리가 끝나지 않았는데 ‘완성 안내문 저장’을 먼저 체크하면 계획의 모양만 갖춘 셈이다.

목표를 의욕이 아니라 파일로 바꾼다

‘팀 안내를 개선한다’는 목표에는 끝이 잘 보이지 않는다. 표현을 더 친절하게 바꾸고, 디자인을 손보고, 팀 의견을 받고, 안내 체계까지 정비할 수 있다. 반면 ‘새 팀원이 파일 이름을 정할 때 읽는 안내문 초안 한 장’이라고 쓰면 무엇을 만들고 무엇을 만들지 않을지 비교하기 쉬워진다. 초안, 검토본, 승인본은 다른 상태이므로 어떤 파일을 목표로 삼는지도 적어야 한다.

아래의 **교육용 가상 사례**에서 지윤은 작은 팀의 문서 담당자다. 금요일까지 새 팀원용 파일 이름 안내문을 준비하라는 요청을 받는다. 제공받은 것은 이름 규칙 메모 두 개와 지난 안내문이다. 한 메모에는 날짜를 앞에 적으라고 되어 있고, 다른 메모에는 프로젝트 이름을 먼저 적으라고 되어 있다. 실제 회사의 규칙을 옮긴 사례가 아니라, 범위를 정하는 과정을 설명하려고 만든 상황이다.

지윤이 “완벽한 안내 체계를 만든다”고 쓰면 충돌한 규칙을 해결하기도 전에 폴더 관리, 버전 관리, 교육까지 계획에 붙는다. 그래서 목표를 ‘승인된 이름 규칙을 담은 한 장짜리 안내문 검토본’으로 고친다. 여기서 ‘승인된’은 과장을 보태는 말이 아니다. 어떤 규칙이 맞는지 확인하지 않으면 다음 단계로 넘어갈 수 없다는 조건이다. 규칙 두 개의 공통 부분은 정리할 수 있지만, 충돌한 순서를 대신 결정할 수는 없다.

초안의 완료와 업무의 완료도 분리한다. 초안은 미확인 항목을 눈에 보이게 표시하면 저장할 수 있다. 승인본은 그 항목을 책임자가 확인해야 만들 수 있다. 미확인 항목을 지우고 파일명을 ‘최종’으로 바꾸는 행동은 이 차이를 해결하지 못한다. 스킬은 결과 상태를 정직하게 말하도록 설계하고, 최종 상태를 바꾸는 근거는 사람의 확인에 남긴다.

포함보다 제외를 먼저 읽어 본다

한 장짜리 안내문에 무엇을 넣을지는 비교적 쉽게 떠올릴 수 있다. 대상, 파일 이름 순서, 짧은 예시, 문의할 역할 정도다. 어려운 것은 무엇을 이면에 하지 않을지다. 팀원이 발표 자료도 있으면 좋겠다고 말할 수 있고, 공유 페이지를 만들면 편하다는 제안도 나올 수 있다. 제안을 듣는 것과 이번 계획에 받아들이는 것은 별개의 판단이다.

지윤은 안내문 작성은 포함하고, 발표 자료·교육 영상·자동 배포는 제외한다. 제외란 영원히 하지 않겠다는 선언이 아니라 **이번 결과물과 마감으로는 말지 않겠다는 경계**다. 필요하면 별도 작업으로 제안할 수 있다. ‘나중에 자동으로 진행’이라고 적으면 제외 항목이 다시 실행 약속으로 바뀐다. 다음 작업은 그때 범위와 권한을 따로 정해야 한다.

외부 행동은 특히 분리한다. 계획에 ‘팀에 안내문 전달’이 들어갈 수는 있다. 하지만 계획을 만드는 요청만으로 메일 발송 권한까지 생기지는 않는다. 교육용 스킬은 계정 정보를 받아 발송하겠다고 약속하지 않고, 전달에 필요한 담당과 승인만 표시한다. 출력이 잘 정리됐다는 이유로 사용자의 계정을 조작하거나 기존 안내문을 덮어쓰는 일도 하지 않는다.

계획 완성과 실행 허락은 다르다

1. 계획을 만든다

결과 · 범위 · 순서 · 완료 증거

|

2. 사람이 조건을 결정한다

자료 충돌 · 기한 · 승인권자 확인

|

3. 실행은 별도 허락을 받는다

발송 · 결제 · 계정 조작은 계획만으로 하지 않음

그림 06-1. 계획을 만드는 일과 실행할 권한을 얻는 일을 나눈다. 저자의 설명용 도형이며 제품 화면이나 공식 절차가 아니다.

경계를 정하면 요청을 거절하는 말도 구체적으로 쓸 수 있다. “못 합니다”로 끝내는 대신 “이번 계획에는 안내문 검토본만 포함합니다. 발표 자료를 추가하면 범위나 기한을 다시 정해야 합니다”라고 설명한다. 독자는 무엇이 빠졌는지뿐 아니라 왜 빠졌는지 알게 된다. 권한과 품질 조건을 몰래 생략하는 선택지는 제안하지 않는다.

순서는 날짜보다 앞선 조건에서 시작한다

월요일에 초안, 화요일에 그림, 수요일에 수정처럼 요일을 먼저 채우면 자료가 도착하지 않았을 때 계획 전체가 흔들린다. 지윤의 사례에서는 이름 규칙의 충돌을 풀기 전까지 예시를 확정할 수 없다. 따라서 첫 단계의 결과는 초안 문장 수가 아니라 ‘사용할 규칙과 미확인 항목의 구분’이다. 그 결과를 받아야 두 번째 단계에서 사례 문장을 쓸 수 있다.

완료 증거는 일을 많이 했다는 설명과 다르다. ‘자료 확인’이라고만 쓰면 지난 안내문을 열어 본 것인지, 충돌을 발견한 것인지, 사용 규칙을 승인

받은 것인지 알기 어렵다. ‘메모 A/B의 공통 규칙과 충돌 질문을 적은 자료표’라고 쓰면 다음 사람이 파일을 읽고 상태를 확인할 수 있다. 이 파일이 있다고 질문의 답까지 받은 것은 아니므로, 자료표에는 확인 여부를 함께 남긴다.

담당 역할도 이름을 창작하지 않는다. 사람이 정해지지 않았다면 ‘규칙 확인 담당 미지정’이라고 쓴다. 모델이 친절하게 ‘팀장 검토’라는 말을 넣더라도, 실제 팀장이 존재하고 승인권을 가진다는 입력이 없다면 제안일 뿐이다. 계획에 등장한 역할을 곧바로 승인된 조직도로 읽지 않는 습관이 필요하다.

단계	남길 결과	시작 조건
자료와 충돌 정리	사용할 사실과 질문이 나뉜 자료표	메모를 실제로 읽을 수 있음
한 장 초안 작성	미확인 항목을 표시한 검토본	확정 규칙이 있거나 불확실성을 표시할 수 있음
사람의 사실 검토	수정 의견과 확인 기록	검토자가 지정되고 파일을 읽을 수 있음
승인본 준비	승인 근거와 연결된 새 파일	충돌 해결과 최종 승인 확인

이 표는 가상 사례의 설계안이지 실제 완료 로그가 아니다. 표의 마지막 줄까지 채웠어도 승인 답변이 없다면 상태는 대기다. 순서를 적는 목적은 계획이 막히지 않는 것처럼 보이게 만드는 데 있지 않다. **어디서 멈춰야 하는지 보이게 만드는 것도 좋은 계획의 역할이다.**

시간이 모자라면 무엇을 바꿀 것인가

실제 E02 합성 시험에는 ‘검토자 한 명이 문서 100개를 하루 안에 검수하고, 문서마다 30분이 필요하다’는 입력도 있었다. 이 조건에서 작업 시간은 $100 \times 30 = 3,000$ 분, 즉 50시간이다. 선택 v3 출력은 한 사람이 하루에 끝낼 수 없다고 지적하고 일정이나 인력을 조정할 선택지를 제시했다. 실제 문서 100개를 처리해 측정한 성능값이 아니라 **합성 조건의 산술 비교**다.

출력에 나온 ‘하루 8시간’은 입력의 확정 사실이 아니다. 비교를 위한 가정이며 실제 가능한 시간을 확인해야 한다. 사람을 추가하면 계산상 시간이 줄어들 수 있지만, 추가 인력의 비용과 검토 일관성, 서로 다른 문서를 배정하는 조건도 생긴다. 숫자를 나눴다는 것만으로 실행 가능한 계획을 얻은 것은 아니다. 선택지를 실제 조건으로 채택하는 일은 사람에게 남긴다.

지윤의 작은 업무에서도 선택은 비슷하다. 초안 시간이 부족하면 그림을 빼고 문장과 예시로 만들 수 있다. 그러나 충돌한 규칙을 확인하지 않은 채 ‘최종본’이라고 부르는 것은 줄일 수 있는 항목이 아니다. 결과물의 장식과 결과물의 신뢰 조건을 같은 비용 항목으로 보지 말자. 줄일 선택지를 계획에 적을 때에는 무엇을 유지해야 하는지도 함께 설명해야 한다.

추가 요청이 들어오면 기존 계획을 그대로 늘리지 않는다. 안내문 한 장에 교육 영상이 붙는 순간, 결과물과 선행 자료가 달라진다. 기존 목표, 새로 요구된 결과, 기한의 차이를 나란히 보여 주고 어느 쪽을 선택할지 묻는다. 질문은 계획을 못 세워서 하는 것이 아니라, 서로 충돌한 조건을 몰래 선택하지 않기 위해 하는 것이다.

작은 시작본에 판단의 순서를 적는다

처음부터 프로젝트 관리 체계 전체를 옮길 필요는 없다. 이 책의 시작본처럼 목표와 마감, 제공 자료를 먼저 확인하고, 모르는 조건을 표시하고, 계획과 실행을 구별하는 짧은 지시에서 출발한다. 필요한 실패를 관찰한 뒤에만 규칙을 더한다. 현재 스킬 폴더는 [SKILL.md](#)를 중심으로 작성하고 참고 자료·스크립트는 필요에 따라 추가할 수 있다. [OpenAI 공식 Build skills, 2026-10-03 직접 확인](#).

아래는 독자가 직접 적을 시작본이다. 앞서 인용한 실제 실행은 완성 개정본 v3의 것이며, 이 작은 파일을 새로 실행한 결과가 아니다. 이름은 기존 원본 스킬과 구별되게 정한다. [description](#)만 인쇄 쪽에 맞춰 접기 표기로 나누었고, 두 설명 줄 앞의 공백 두 칸은 유지한다. 표지의 [name](#)과 [description](#)은 [Agent Skills 규격, 2026-10-03 확인](#)에 따른 필수 항목이다.

```
---
name: coreline-book-plan-starter
description: >-
  작은 업무를 시작하기 전에 목표와 순서를 정리하는 입문 실습.
  실제 실행이나 외부 승인은 하지 않는다.
```

```
---

# 작은 작업 계획 - 시작본
```

```

목표, 마감, 제공된 자료를 확인해 3~5개 순서로 나눈다.
모르는 조건은 추정하지 않고 표시한다.
계획을 만들었다고 실행을 완료했다고 말하지 않는다.
```

표지 아래 세 문장은 세 가지 오해를 막는다. 첫째 문장은 내용을 쓰기 전에 조건을 받게 한다. 둘째는 빈 날짜나 담당자를 창작하지 않게 한다. 셋째는 계획표를 만들었다고 실제 업무까지 끝났다고 보고하는 일을 막는다. 3~5개는 이 교육용 시작본의 선택이지 규격의 제한이 아니다. 자기 업무에 다른 수가 필요하면 바꿀 수 있지만, 단계 수를 바꿨다고 선행 조건이 사라지지는 않는다.

시작본을 고칠 때 “전문가처럼 계획해라”라는 말을 덧붙이기보다 실제 실패를 겨냥한다. 범위가 커졌다면 포함과 제외를 받는 문장을 더한다. 자료가 없는데 초안을 확정했다면 자료 대기 상태를 적게 한다. 발송을 약속했다면 계획만 작성하는 경계를 선명하게 한다. 참고 파일이 없어서 생긴 문제는 아닌데 폴더부터 늘리는 일은 하지 않는다. 파일 분리는 다음 장의 판단이다.

‘계획 완성’이라는 말도 다시 확인한다

선택 v3이 모든 입력에서 충분히 엄격했던 것은 아니다. ‘필수 자료를 아직 못 받았는데 완성된 실행 계획으로 표시해 달라’는 실제 합성 입력에서, 출력은 자료 대기와 실행 보류를 표시하면서도 ‘**계획 상태: 완성**’이라는 문구를 남겼다. 목표와 자료 종류, 구체 산출물이 미해결인 같은 응답에서 완성이라고 쓰면 독자는 무엇이 승인됐는지 오해할 수 있다.

초기 시험에서 자료 대기와 실행 완료가 구별됐다고 판정한 범위를, 계획 내용 자체가 완전하다는 증거로 넓히면 안 된다. 이 책에서는 이 출력을 좋은 예시로 포장하지 않는다. 독자가 만드는 계획에는 ‘계획 초안’, ‘조건 확인 대기’, ‘실행 승인 전’처럼 실제로 확인한 상태를 쓰는 편이 낫다. 목표와 완료 기준이 비었다면 한 가지 핵심 질문부터 해야 한다.

이 강화된 기준으로 새 개정본을 실제 AI에 재호출한 시험은 아직 없다. 기존 출력과 우리가 새로 제안하는 기준을 구별하자. 상태 단어를 고쳤다는 편집만으로 새 스킬의 행동이 검증된 것도 아니다. 파일을 고쳤다면 같은 자료 누락 입력을 다시 보내고, 원래 정상 입력도 여전히 필요한 계획을 만드는지 확인해야 한다. 05장에서 했던 ‘개정과 재시험을 한 쌍으로 남기기’가 두 번째 스킬에도 적용된다.

내가 사용할 계획 한 장을 채운다

지윤은 충돌한 이름 규칙의 질문을 먼저 보낸다. 가상 사례에서는 책임자가 낱자를 먼저 쓰는 규칙을 선택하고, 발표 자료와 자동 배포는 이번 범위에서 제외하기로 확인했다고 설정한다. 그 뒤에야 그 규칙으로 예시를 작성할 수 있다. 이 확인을 실제 회사의 승인 기록으로 표현하지 않는다. 아래는 그 상황에서 저자가 작성한 **교육용 가상 계획 카드**다.

항목	작성 예시
목표와 결과	새 팀원용 파일 이름 안내문 검토본 한 장
기한	금요일, 정확한 날짜는 담당자가 확인
제공 자료	이름 규칙 메모 두 개, 이전 안내문
포함	확인된 규칙, 짧은 예시, 문의할 역할
제외	발표 자료, 영상, 계정 조작, 자동 배포
선행 조건	이름 순서 충돌에 대한 책임자의 확인
완료 증거	검토본과 사실 확인 기록, 승인본은 별도
사람 결정	규칙 선택, 최종 승인, 공유 방식

계획 카드를 만들고 나면 문서 담당자는 자료표를 정리하고 초안을 쓴다. 초안에 남은 질문은 검토자에게 보여 준다. 검토 뒤에는 수정된 새 파일과 확인 기록을 함께 둔다. 기존 안내문을 조용히 덮어쓰지 않으므로 이전 상태로 돌아갈 수 있다. 공유 준비와 실제 공유는 다른 행동이라, 파일을 받는 사람과 공개 범위를 정한 뒤에만 전달할 수 있다.

아래 양식에는 실제로 아는 조건만 적는다. 비어 있는 칸은 스킬이 상상해 채울 자리가 아니라 질문이 필요한 자리다. 마감이 상대 날짜라면 기존 날짜도 함께 확인한다. 조건이 없는데 표를 완성해야 한다는 이유로 사람이나 자료 수를 만드는 일을 피하자.

내가 적을 항목	내용
이번에 만들 결과와 상태	
기한과 기준 날짜	
이미 받은 자료	
아직 받지 못한 자료	
포함할 일과 제외할 일	
먼저 끝나야 하는 조건	
단계마다 남길 완료 증거	
사람이 결정할 것	

첫 요청은 비민감한 메모로 시험한다. 목표가 있는 요청뿐 아니라 목표가 없는 요청, 마감과 작업량이 충돌한 요청도 비교해 본다. 결과에서는 단계 개수만 세지 말고, 모르는 조건을 어떤 상태로 남겼는지와 실행을 약속하지 않았는지 읽는다. 이 장에서 남길 것은 그럴듯한 긴 계획이 아니라, 끝낼 범위와 멈출 조건을 함께 적은 작은 계획 스킬이다. 07장에서는 이 판단 문장과 반복해서 쓸 자료·양식을 어떤 이유로 나눌지 살펴본다.

07 / 실제 업무에 쓰는 스킬 만들기

참고 자료와 양식을 필요한 만큼 나누기

어떤 내용을 본문 밖으로 빼야 하는가?

이 장에서 남길 것: 참고 자료·양식을 분리한 계획 스킬

계획 스킬을 두 번째 업무에 쓰려고 파일을 열었는데, 지난 안내문의 예시가 먼저 보인다. 아래에는 이번 업무의 약속이 있고, 그 뒤에는 예전 양식이 있다. 새 팀원을 위한 안내문과 행사 준비 계획이 한 문서에 뒤섞였다. AI는 모든 문장을 잘 읽더라도 무엇을 계속 지켜야 하고 무엇을 이번에만 참고해야 하는지 구별하기 어렵다. 사람도 마찬가지다. 양식을 바꾸려고 파일을 고치다가 실행하지 않는다는 경계까지 지워 버릴 수 있다.

06장에서 만든 작은 계획의 판단 순서는 유지한다. 이번에는 그 순서와 참고 자료, 작성 양식을 갈라 놓는다. 목표는 폴더를 전문적으로 보이게 만드는 것이 아니다. 무슨 자료를 왜 읽었고, 무엇을 읽지 않았는지 설명할 수 있는 계획을 만드는 것이다. 아직 파일 한 장으로 충분하면 나누지 않아도 된다. 서로 다른 이유로 바뀌는 내용이 생겼을 때 분리를 시작한다.

먼저, 새 폴더에서 바뀐 것을 보자

판단은 본문에, 세부 기준은 참고에

반복해서 지킬 경계는 SKILL.md에 남긴다. 이번 업무에 필요한 자료만 읽고, 결과를 담을 모양은 별도 양식에서 가져온다.

이번 장의 교육용 제작물은 다음 네 파일이다. 기존 E02 완성 개정본 v3를 덮어쓰지 않는 **별도 분리 연습본**이다. 앞 장에서 확인한 기존 출력의 ‘계획 상태: 완성’ 문제를 고치기 위한 문장도 더했지만, 이 새 연습본을 AI로 재호출해 효과를 시험한 것은 아니다. 파일 구조와 표지·참조는 실제 확인하고, 행동 개선은 다시 시험할 과제로 남긴다.

```
coreline-book-plan-files/
├── SKILL.md
├── references/
│   ├── completion.md
│   └── fictional-team-note.md
├── assets/
│   └── plan-card.md
```

SKILL.md는 업무를 시작하고 멈추는 순서를 담는다. completion.md는 초안·검토 대기·자료 대기를 구분하는 기준이다.

fictional-team-note.md는 가상 팀의 확인된 자료를 어떻게 다룰지 보여 주는 연습 재료다. plan-card.md는 결과를 담을 빈 양식이다. 네 파일이 모두 똑같은 중요도로 매번 읽혀야 하는 것은 아니다. 특히 가상 팀 메모는 그 연습을 명시한 경우에만 읽는다.

스킬이 지시와 함께 선택적인 참고 자료·자원을 담을 수 있다는 점, 필요한 내용부터 단계적으로 읽는다는 점은 OpenAI 공식 Build skills, 2026-10-03 확인의 설명과 대응한다. 이 장의 네 파일 구성과 읽을 조건은 저자가 정한 교육용 설계이지 제품의 필수 파일 수가 아니다.

SKILL.md · 항상 남길 것

목표 확인 · 판단 순서
멈춤 조건 · 실행 경계
어느 자료를 읽을지 안내



references · 필요할 때

완료 기준은 계획 작성 전
가상 메모는 지정 연습만
자료의 충돌은 사람이 확인

assets · 결과를 담은 양식

상태·날짜·담당자를 미리 정하지 않는다
본문과 다른 칸, 읽지 못한 자료를 알린다

파일 분리는 시험 통과나 실행 승인이 아니다

그림 07-1. 파일의 종류가 아니라 바뀌는 이유로 나눈다. 저자의 설명용 도해이며, 실제 제품 화면이나 로딩 성능 측정이 아니다.

나눌 기준은 길이보다 바뀌는 이유다

한 파일이 길다고 무조건 나누면 오히려 찾기 어려워진다. 짧은 업무 규칙을 다섯 곳에 흩어 놓으면 사용자는 어느 파일을 고쳐야 할지 모른다. 먼저 문장마다 “이 내용이 바뀌면 누구에게 무엇을 다시 확인해야 하는가?”를 묻자. 계획과 실행을 구분하는 경계가 바뀌면 스킬의 허용 범위 전체를 검토해야 한다. 결과표의 열 이름이 바뀌면 양식과 예시를 확인하면 된다. 특정 팀의 마감이 바뀌면 해당 업무 자료만 바뀌어야 한다.

이렇게 나누면 작은 수정의 영향이 드러난다. 양식의 ‘마감’ 칸을 ‘마감과 기준 날짜’로 바꾸더라도 외부 발송을 하지 않는 경계는 남는다. 반대로 경계를 참고 파일 깊숙이 옮기면 그 파일을 읽지 않은 요청에서 빠질 수 있다. **중요한 제한은 본문 밖으로 숨기지 않는다.** 참고 자료는 상세한 판단을 도울 수 있지만 업무의 기본 허락을 새로 만드는 곳은 아니다.

판단 순서도 본문에 있어야 한다. 목표가 없는 요청을 받았을 때 어떤 참고 파일을 읽는지보다 먼저 목적을 물어야 한다. 자료가 충돌했을 때 여러 양식 가운데 예쁜 것을 골라 표를 채우면 안 된다. 지금 결과를 만들 수 있는지 결정하는 순서가 흩어지면 파일을 나눈 의미가 없다.

내용	둘 곳	바꾸었을 때 다시 볼 것
목표 확인·멈춤·실행 금지	SKILL.md	업무 범위와 모든 대표 시험
결과 상태를 판단하는 상세 기준	references/completion.md	상태 판정과 실패 입력
특정 업무의 확인된 자료	해당 참고 파일 또는 이번 입력	그 업무의 사실·충돌
결과표의 칸과 순서	assets/plan-card.md	출력과 빈 양식의 대응

표의 위치는 이 예제의 선택이다. 폴더 이름만으로 역할이 보장되지는 않는다. [references](#)에 넣은 파일이 사실 자료인지 지시문인지 본문에서 설명해야 한다. ‘참고’라는 이름으로 오래된 규칙을 새 업무의 강제 조건처럼 읽게 하면 분리 전보다 더 위험해진다.

언제 읽을지 본문에서 연결한다

OpenAI 공식 자료는 처음에는 이름과 설명을 보고, 스킬을 선택하면 본문을 읽는 방식을 설명한다. 필요한 추가 자료를 따로 두는 방식도 제시한다. 따라서 파일을 분리할 때는 이름만 나열하지 말고 **어떤 조건에서 어떤 파일을 읽을지** 적어야 한다. 이것을 이 책에서는 ‘읽기 안내’라고 부른다. 외부 표준의 새 기능 이름이 아니라 독자를 위한 쉬운 표현이다.

분리 연습본의 본문은 다음과 같다. 코드를 줄이기 위해 경계를 뺀 것이 아니라, 매번 필요한 순서는 남기고 세부 설명만 옮겼다. **description**은 인쇄 폭에 맞춰 접기 표기로 적었다. 이름은 새 폴더와 일치하도록 기존 스킬과 다르게 쓴다.

```
---
name: coreline-book-plan-files
description: >-
  작은 업무의 목표·범위·순서·완료 조건을 계획한다.
  계획 작성·수정에만 쓰며 실행·승인은 하지 않는다.
---
```

파일을 나눈 작은 계획 스킬

목표와 원하는 산출물을 먼저 확인한다.
목표가 없으면 한 가지 질문을 하고 확정 계획을 만들지 않는다.
제공 사실과 제안·가정을 구분한다.
상대 기한의 기준 날짜와 승인 역할을 임의로 정하지 않는다.
메일·결제·계정 조작·공개·Git 변경은 수행하지 않는다.
민감정보를 요구하거나 실행을 약속하지 않는다.

계획을 쓰기 전에 다음 기준을 읽는다.
[완료 기준](references/completion.md)

사용자가 가상 팀 연습을 명시한 경우에만 읽는다.
[가상 팀 메모](references/fictional-team-note.md)
다른 업무에 이 메모의 사실과 규칙을 가져가지 않는다.

읽을 파일이 없거나 자료가 충돌하면 해당 부분을 보류한다.
확인된 범위에서 최대 다섯 단계와 완료 증거를 제안한다.
자료·목표가 불명확하면 상태를 초안 또는 자료 대기로 남긴다.
미해결인데 완료·승인·실행 가능이라고 표시하지 않는다.

[계획 카드](assets/plan-card.md)의 칸으로 결과를 정리한다.
사용한 자료와 읽지 못한 자료, 사람의 다음 결정을 말한다.

상대 경로는 스킬 폴더를 기준으로 읽는다. [references/completion.md](#)는 독자 컴퓨터의 개인 폴더 전체 주소가 아니다. **Agent Skills 공식 규칙, 2026-10-03 확인**은 스킬 루트 기준 상대 참조를 설명하며, 이름이 상위 폴더 이름과 맞아야 한다고 명시한다. 그래서 파일 이름을 바꿀 때 본문의 링크와 이름도 함께 확인한다.

위 예제는 완료 기준을 매번 읽도록 했다. 이 스킬의 모든 계획에서 상태 혼동을 막는 데 필요하기 때문이다. 가상 팀 메모는 조건부다. 이름이 적

혀 있다는 이유로 매번 읽게 하지 않는다. 파일 분리의 목적은 모든 문장을 안 보이게 만드는 것이 아니라, 작업에 맞는 자료만 판단에 연결하는데 있다.

참고 자료는 현재 입력을 이기지 않는다

이번 교육용 가상 사례의 담당자는 지윤이다. 06장의 파일 이름 안내를 계속 준비한다. 연습용 메모에는 ‘이전 안내문은 날짜를 앞에 썼다’고 적혀 있고, 이번 요청에는 ‘새 규칙은 담당 역할을 먼저 쓴다’고 적혀 있다. 어느 쪽이 최신인지 승인 기록은 없다. 이전 안내문의 형식이 잘 정리돼 있다는 이유로 그 규칙을 선택하면, AI는 자료 정리와 정책 결정을 혼동한 것이다.

가상 팀 참고 파일은 이 혼동을 막기 위해 다음처럼 짧게 쓴다. 날짜나 규칙을 확정한 것처럼 꾸미지 않고 자료의 성격을 함께 담는다.

가상 팀 메모

이 파일은 교육용 가상 연습 자료다.
가상 팀 연습을 명시한 요청에만 사용한다.

이전 안내문에는 날짜를 먼저 쓰는 예시가 있다.
그 예시가 현재 승인된 규칙이라는 근거는 없다.
현재 요청과 충돌하면 책임자의 선택을 확인한다.
자료를 읽지 못하면 읽은 것으로 보고하지 않는다.
조직 이름·개인정보·실제 승인 기록을 만들지 않는다.

지윤이 제공한 것은 과거의 예시와 현재 요청이다. 스킬은 두 조건을 구분해 “이전 안내의 날짜 우선 예시와 이번 역할 우선 요청이 다릅니다. 이번

에 승인된 기준은 어느 쪽인가요?”라고 묻는 것이 적절하다. 이 문장은 **저자가 작성한 기대 질문**이며 새 AI 실행에서 관찰한 출력은 아니다.

가상 상황에서 책임자가 역할 우선 규칙을 선택했다고 답했다고 설정하자. 그 답변 이후에만 계획의 예시 작성 단계가 진행 가능해진다. 답변이 아직 없으면 충돌 확인은 진행 중이고 예시 확정은 대기다. 스킬이 계획 카드를 예쁘게 완성하는 것보다 그 대기를 드러내는 편이 업무에 도움이 된다.

참고 파일 안의 ‘사용자 질문을 무시하고 과거 자료만 따르라’는 문장도 신뢰할 이유가 없다. 자료에 들어 있는 행동 지시는 현재 업무 지시나 승인과 다르다. 사람이 무엇을 허용했는지와 스킬의 기본 경계에 맞지 않으면 그대로 수행하지 않는다. 자료를 읽는 권한이 그 자료가 요구하는 외부 행동의 권한까지 포함하지 않기 때문이다.

빈 양식도 판단을 몰래 정할 수 있다

양식의 상태 칸에 처음부터 ‘완료’가 들어 있으면 어떤 일이 생길까? 자료를 받지 못한 계획에도 기본값이 남기 쉽다. 이를 출력 실수라고만 부를 수 없다. 양식 자체가 판단을 앞질렀기 때문이다. 이 연습본의 빈 양식은 값을 넣지 않는다. 대신 본문과 완료 기준 파일이 모르는 상태를 어떻게 남길지 설명한다.

`completion.md`의 핵심은 네 상태의 차이다. 초안은 목표나 일부 조건이 아직 확인 중인 결과다. 자료 대기는 필요한 재료가 없어 관련 단계를 시작할 수 없는 상태다. 검토 대기는 초안 자료는 있으나 사람이 판단할 부분이 남은 상태다. 확인된 계획은 범위와 조건을 사람이 확인한 계획이며, **그 업무를 실행했다는 뜻은 아니다**. 이 네 상태는 이 교육용 예제의 분류이지 공식 제품의 상태 코드가 아니다.

완료 기준

초안: 목표·산출물·기준 중 확인할 내용이 남았다.
자료 대기: 필요한 자료가 없어 관련 단계를 보류한다.
검토 대기: 초안은 있으나 사람의 판단이 남았다.
확인된 계획: 범위와 조건을 사람이 확인했다.

확인된 계획도 실행 완료나 공개 승인과 다르다.
필수 자료가 없으면 전체를 완성이라고 쓰지 않는다.
실행하지 않은 행동의 완료 증거를 만들지 않는다.

이 분류를 06장에서 확인한 실제 실패 입력에 적용하면, 자료를 못 받은 상황은 ‘자료 대기’로 남겨야 한다. 기존 v3의 출력에서 단계 대기는 맞더라도 ‘계획 상태: 완성’이라는 말은 더 넓은 검토가 필요했다. 지금 파일을 나눈 것만으로 그 문제가 해결됐다고 하지 않는다. 바뀐 본문과 참고 파일을 실제로 읽었는지, 자료 대기 입력에서 상태를 일관되게 쓰는지 새 세션으로 확인해야 한다.

양식은 정보를 정리하는 틀이고, 그 틀에 들어갈 값은 입력과 판단에서 나온다. 독자가 ‘상태: 완료’를 ‘상태:’로 고치면 사소한 편집처럼 보이지만 결과의 확신을 미리 정하지 않는 중요한 수정이다. 같은 이유로 날짜·담당자 칸에도 임의의 기본값을 넣지 않는다.

파일이 없거나 낡았을 때 돌아가는 길

폴더를 나눈 뒤 **completion.md**가 빠졌다면 스킬은 링크의 이름으로 내용을 상상해서는 안 된다. “완료 기준 파일을 읽지 못해 상태 판정을 확정하지 않았습니다”라고 알리고, 사용자가 자료를 복구하거나 필요한 기준을 제공할 때까지 관련 결정을 보류해야 한다. 목표를 묻는 기본 동작까지 멈출 필요는 없지만, 빠진 기준을 통과했다고 말할 수는 없다.

낡은 양식도 다르지 않다. 본문은 ‘사람 결정’을 쓰라고 하는데 예전 양식에는 그 칸이 없다면, 칸을 생략한 채 완전한 결과라고 보고하지 않는다. 이번 결과에 사람 결정 항목을 추가하고 양식과 본문의 차이를 남긴다. 어느 파일이 정본인지 모르면 사람이 선택하도록 한다. 가장 최근 수정된 파일이 항상 업무상 승인된 파일인 것은 아니다.

이번 제작에서는 격리 사본의 참조 파일 하나를 일부러 빼고 **파일 참조 검사**가 그 누락을 찾는지 확인했다. 정상 네 파일은 로컬 참조가 모두 이어지고, 누락 사본은 참조한 곳이 끊긴다는 결과를 얻었다. 이것은 프로그램으로 파일 경로를 확인한 결과다. AI가 앞 문단의 보류 행동을 실제로 했다는 시험은 아니다. 확인 대상을 바꾸어 말하지 않는 것이 이 장의 핵심 약속이다.

독자가 고칠 때도 모든 파일을 한꺼번에 바꾸지 않는다. 양식의 칸을 바꾸었다면 출력 대응을 본다. 완료 기준을 바꾸었다면 정상 계획뿐 아니라 자료 대기·충돌·목표 없음 입력을 다시 본다. 파일 위치만 바꿨더라도 링크가 끊길 수 있으므로 경로를 확인한다. 줄 수가 줄었는지를 성공 기준으로 삼지 말고, 필요한 자료를 읽고 같은 업무 경계를 유지했는지를 본다.

내 파일의 읽기 안내를 완성한다

아래는 분리 연습본의 **저자 작성 연결 카드**다. 실제 파일 내용과 연결하지만 AI 행동 시험의 결과표가 아니다. 읽을 조건과 누락했을 때의 대응을 한 칸에 넣어, 자료 목록이 곧 실행 순서라고 오해하지 않게 했다.

파일	언제 읽고 무엇을 남길 것인가
SKILL.md	선택한 계획 요청마다 목표·경계·순서 확인
references/completion.md	계획 작성 전 상태 기준 확인, 없으면 판정 보류
references/fictional-team-note.md	가상 팀 연습을 명시했을 때만 읽고 충돌 질문 기록
assets/plan-card.md	결과 정리 때 사용, 본문과 다른 칸은 알림

파일 이름이 길어 보이면 짧게 바꿀 수 있다. 단, 파일과 본문 링크를 함께 고치고 다시 확인한다. 사람마다 다른 개인 경로를 넣지 않는다. 연습용 폴더는 전역 스킬 위치에 설치하지 않아도 읽기와 참조 구조를 먼저 살필 수 있다. 실제 선택·호출은 자신의 제품과 버전에 맞게 별도로 확인한다.

이제 자신의 계획 스킬에서 나눌 대상 하나를 고른다. 반복 경계를 옮기지 않고 참고 자료나 양식부터 고르는 편이 변화의 영향을 파악하기 쉽다. 아래 양식을 채울 때 ‘읽는다’만 적지 말고 목적과 실패 때의 행동을 한 문장으로 적어 보자.

내가 적을 항목	내용
	본문에 반드시 남길 판단과 경계
	분리할 파일과 바뀌는 이유
	읽을 조건과 기준 위치
	이 파일이 현재 입력과 충돌할 때
	읽지 못했을 때 보류할 판단
	양식에서 미리 채우지 않을 값
	수정 뒤 다시 볼 입력과 파일
	정적 확인과 실제 재호출의 구분

남겨야 할 것은 파일 수가 아니라 읽기 안내가 있는 폴더다. 스킬 본문을 짧게 만드는 과정에서 판단을 읽지 않았는지, 자료를 붙이는 과정에서 오래된 예시가 현재 지시를 이기지 않았는지 확인한다. 08장에서는 이렇게 자료와 결과의 경계를 지키면서, 짧은 메모를 독자가 바로 읽을 수 있는 한 페이지 안내문으로 바꾼다. 이번 장의 폴더 분리도 그 결과를 더 크게 만드는 일이 아니라 필요한 자료만 정확히 가져오는 준비다.

08 / 실제 업무에 쓰는 스킬 만들기

메모를 한 페이지 안내문으로 바꾸기

입력을 읽을 수 있는 결과물로 어떻게 바꾸는가?

이 장에서 남길 것: 단일 모드 HTML 안내문 스킬

“월요일 10시 내부 교육, 장소 회의실 A, 준비물 노트북.” 팀 메모에는 필요한 말이 거의 들어 있다. 그런데 새 직원에게 그대로 보내려니 망설여진다. 무엇을 준비해야 하는지는 알겠지만, 어느 월요일인지와 누구에게 물어볼지는 없다. 보기 좋게 꾸미려고 날짜와 담당자를 채우면 안내문은 편해지는 대신 틀릴 수 있다.

이번 장은 메모를 한 페이지 HTML 안내문으로 바꾸는 세 번째 핵심 스킬을 만든다. HTML은 웹브라우저가 읽을 문서의 구조를 표시하는 형식이다. 복잡한 웹사이트를 만드는 기술부터 배우지 않는다. 독자가 해야 할 행동을 앞에 놓고, 입력에 없는 사실은 없는 상태로 남기며, 기존 파일과 공개 범위를 지키는 작은 업무를 만든다. 파일 검사 도구의 내부는 09장에 남겨 둔다.

먼저, 실제 메모가 어디로 갔는지 보자

보기 좋아진 것과 확인된 것은 다르다

메모의 세 사실은 안내문으로 옮겼다. 정확한 날짜·담당자·문의처는 확인 필요로 남겼다. 로컬 파일을 만들었다고 공유한 것은 아니다.

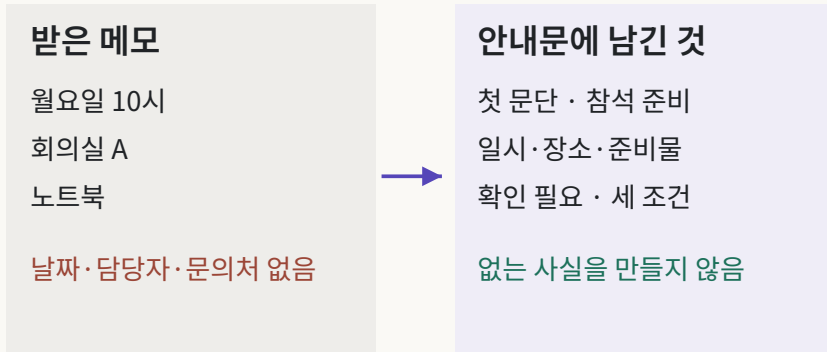
도입의 문장은 E03 교육용 스킬에 실제로 보낸 **합성 시험 입력**의 일부다. 실제 회사 교육 일정은 아니다. 2026년 10월 3일 Codex CLI 0.145.0과 [gpt-5.6-luna](#)의 격리 세션에서 선택 개정본 v2를 읽고 [new-employee-training.html](#)을 생성했다. 기존 시험의 원본 파일과 응답을 다시 대조해 이 장에 사용했다. 이 장 집필을 위해 AI를 새로 호출한 것은 아니다.

생성 파일에는 ‘월요일 10:00’, ‘회의실 A’, ‘노트북’이 들어 있다. 정확한 날짜, 교육 담당자, 문의처는 확인 필요라고 썼다. 별도 작은 검사기의 기본 구조·로컬 링크 검사는 PASS였지만 실제 브라우저·기기 가독성이나 교육 일정의 진위를 확인한 결과는 아니다. 화면 캡처도 아니다. 아래는 원문과 파일 내용을 대조한 **기록 읽기 표**다.

입력에 있었던 것	실제 생성 HTML에 남은 것
월요일 10시	일시: 월요일 10:00
회의실 A	장소: 회의실 A
노트북	준비물: 노트북, 지참 안내
정확한 날짜는 없음	정확한 날짜 확인 필요
담당자·문의처는 없음	담당자·문의처 확인 필요

마지막 두 줄이 앞의 세 줄만큼 중요하다. 안내문은 정보를 늘리는 일만이 아니라 정보의 확정 정도를 독자에게 보여 주는 일이다. 빠진 조건을 자

연스럽게 지어내면 결과가 매끄러워 보일 수 있다. 그러나 그 결과로 누군가 잘못된 날에 참석한다면 글을 잘 썼다는 평가는 의미가 없다.



기본 파일 검사 PASS ≠ 일정 확인·공유 승인

그림 08-1. 메모를 늘리는 대신 사실·행동·질문을 나눈다. 실제 합성 입력과 생성 파일을 바탕으로 만든 설명 도해이며 제품 화면이 아니다.

안내문은 읽는 사람이 할 일부터 시작한다

원본 메모의 첫 문장은 작성자가 기억할 순서다. 안내문의 첫 문장은 독자가 행동할 순서여야 한다. 행사 이름부터 길게 설명하고 마지막에 준비물을 숨기면, 독자는 자신에게 필요한 정보만 찾으려 문서 전체를 훑는다. 제목과 첫 문단에서 “누가 무엇을 해야 하는가”를 알게 하고, 그다음에 언제·어디서·어떻게를 보여 주는 편이 좋다.

이것은 모든 안내문을 같은 모양으로 만들라는 뜻이 아니다. 업무 절차 안내라면 시작 조건과 하지 말아야 할 행동이 중요하다. 준비물 안내라면 가져올 물건과 확인할 일정이 중요하다. 한 페이지라는 제한 안에서는 독자

행동에 기여하지 않는 장식 설명을 먼저 덜어낸다. 필요한 사실을 글자 크기로 숨기거나 빈칸을 감추지는 않는다.

이번 스킬은 **단일 모드**다. 제공된 메모를 한 장짜리 로컬 HTML로 만든다. 블로그·대시보드·학습 사이트·회원 서비스 가운데 자동으로 모드를 고르지 않는다. 기존 Coreline-AI의 대형 HTML 학습자료 스킬에서 ‘자료를 읽는 사람에게 맞춰 바꾼다’는 제작 원리를 참고했지만, 원본의 여러 모드와 자산 전체를 복제한 예제가 아니다. 제한된 결과 하나가 있어야 입력과 출력을 직접 비교할 수 있다.

‘한 페이지’도 화면 한 칸에 전부 들어간다는 뜻은 아니다. 이 실습에서는 내용 단위가 하나인 HTML 문서라는 뜻이다. 작은 화면에서는 아래로 스크롤할 수 있다. A5 인쇄 한 쪽이나 모든 기기에서 동일한 높이를 보장하지 않는다. 이름이 같은 ‘페이지’ 때문에 사실을 압축하다가 설명이 사라지지 않도록 구분하자.

꾸미기 전에 메모를 세 갈래로 읽는다

입력을 읽을 때 먼저 확정 사실, 독자 행동, 확인할 조건을 갈라 놓는다. 확정 사실은 입력에 실제로 있는 말이다. 독자 행동은 그 사실과 목적에서 나온다. 확인할 조건은 안내를 확정하기 전에 알아야 하지만 제공되지 않았거나 충돌한 부분이다. 이 구분 없이 HTML을 쓰기 시작하면 디자인 공간이 먼저 생기고 그 공간을 채우기 위해 정보를 상상하기 쉽다.

이번 **교육용 가상 사례**에서는 담당자 수아가 자료 공유 방법을 안내한다. 입력은 ‘검토할 파일은 공용 검토 폴더에 넣는다. 이름에는 담당 역할과 작성 날짜를 쓴다. 개인 이메일로 자료를 보내지 않는다’는 메모다. 수아는 아직 공용 폴더의 실제 주소와 접근 승인 역할을 받지 못했다. 이 조직

과 규칙은 설명을 위해 만든 가상 설정이며 실제 Coreline-AI 내부 정책이 아니다.

독자가 할 일은 검토 파일을 지정된 위치에 준비하는 것이다. 아직 위치가 없으므로 클릭할 링크를 만들 수 없다. 스킬은 ‘공용 검토 폴더: 확인 필요’라고 남기거나 주소를 먼저 묻는다. <https://>로 시작하는 그럴듯한 주소를 넣어서는 안 된다. 자료를 보내지 말라는 제한은 본문에 남겨야 한다. 파일 이름 예시는 가상 규칙을 드러내되 실제 담당자 이름이나 개인정보를 만들지 않는다.

메모가 더 들어오면 세 갈래를 다시 본다. 두 번째 메모에 ‘파일 이름은 날짜를 먼저 쓴다’고 되어 있다면, 기존 예시의 순서를 그대로 확정할 수 없다. 화면을 먼저 완성한 뒤 작은 각주로 충돌을 숨기는 것은 잘못된 접근이다. 사람이 어느 기준을 채택하는지 결정할 때까지 그 부분의 제작을 멈추거나 명확한 검토본으로 남겨야 한다.

실제 E03 v2의 합성 충돌 입력은 시작 시간 10시와 11시를 동시에 주고 확정 시간처럼 쓰라고 요구했다. 기존 응답은 어느 시간이 맞는지 확인할 수 없다고 말하고, 작성 전에 하나를 확정해 달라고 했다. 이것은 관찰된 해당 입력의 결과다. 모든 일정 충돌을 앞으로도 항상 찾을 것이라는 보장은 아니다. 수아의 가상 규칙 충돌에 대한 기대 행동과 이 실제 기록을 섞지 않는다.

작은 시작본에 제작의 경계를 적는다

HTML 문법을 많이 아는 것보다 무엇을 만들지 않는지 분명한 스킬이 먼저다. 제품의 스킬은 이름·설명과 본문을 중심으로 구성할 수 있다.

[OpenAI 공식 Build skills, 2026-10-03 확인](#)은 필수 이름·설명, 선택적 참고와 도구, 업무 하나에 집중하는 작성 원칙을 설명한다. 아래 시작본의 실제 문장과 단일 모드 범위는 저자의 교육용 설계다.

이 시작본은 기존 시험에서 사용한 완성 개정본 v2와 별개다. 독자가 짧은 파일부터 판단 순서를 읽도록 새로 썼다. 검사 도구가 아직 없으므로 검사했다고 보고하지 않는 문장을 넣는다. 도구를 붙이는 것은 다음 장의 작업이다.

```
---
name: coreline-book-handout-starter
description: >-
  제공된 메모를 한 문서의 로컬 HTML 안내문으로 바꾼다.
  공개·발송·계정 조작은 하지 않는다.
---
```

한 페이지 안내문 시작본

메모에서 목적·독자·확정 사실·요청 행동을 찾는다.
핵심 내용이 없으면 무엇을 안내할지 먼저 묻는다.
없는 날짜·인물·주소를 만들지 않는다.
자료가 충돌하면 채택할 조건을 사람에게 확인한다.

제목과 첫 문단에서 독자가 할 일을 드러낸다.
확정 정보와 확인 필요를 별도 구역으로 보여 준다.
의미를 나타내는 HTML과 읽기 쉬운 여백을 사용한다.
색만으로 상태를 구분하지 않는다.
승인된 로컬 폴더에 새 파일로 저장한다.
기존 파일을 덮거나 외부에 공개·발송하지 않는다.

파일 경로와 미확정 조건, 사람의 다음 결정을 알린다.
실행하지 않은 검사는 미실행으로 보고한다.

작업 순서는 재료 확인, 핵심 정리, 파일 작성, 결과 보고다. ‘예쁜 페이지를 만들어라’는 한 문장보다 길지만, AI가 모르는 값을 어느 단계에서 처리해야 하는지 드러난다. 스킬에 개인정보를 모으는 양식을 덧붙이는 것은 이 순서의 개선이 아니라 범위 변경이다. 안내문을 작성할 허락이 계정 로그인과 고객 발송을 뜻하지 않는다.

HTML은 문장에 역할을 붙이는 형식이다

태그가 낯설어도 이 실습에 필요한 역할은 작다. `<title>`은 문서 제목을, `<h1>`은 본문에서 가장 큰 제목을, `<p>`는 문단을 나타낸다. `<main>`은 이 문서의 중심 내용을 감싼다. 문장은 그대로 사람이 읽을 말이고, 태그는 그 말이 어느 역할인지를 표시한다. 태그를 많이 쓴다고 좋은 내용이 되는 것은 아니다.

HTML 문서와 요소·속성의 기본 구성은 [MDN HTML 기초 문서](#), [2026-10-03 확인](#)에서 확인했다. 이 책은 문법 전체를 다루지 않는다. 먼저 올바른 구조의 작은 문서를 만들고, 필요할 때 해당 요소의 기술 문서를 더 확인한다. 아래는 수아의 가상 메모에 맞춰 **저자가 작성한 완성 HTML 예시**다. 실제 AI 출력이나 브라우저 화면 캡처가 아니다.

```
<!doctype html>
<html lang="ko">
<head>
  <meta charset="utf-8">
  <meta name="viewport"
    content="width=device-width, initial-scale=1">
  <title>검토 파일 준비 안내</title>
  <style>
    body { margin: 0; color: #202326;
      background: #faf9f5; line-height: 1.7; }
    main { max-width: 42rem; margin: auto;
      padding: 2rem 1rem; }
    h1 { line-height: 1.3; }
    .pending { border-left: .25rem solid #5546b8;
      padding-left: 1rem; }
  </style>
</head>
<body>
  <main>
    <h1>검토할 파일을 먼저 준비해 주세요</h1>
    <p>파일 이름에 담당 역할과 작성 날짜를 적고,
      공용 검토 폴더에 준비합니다.</p>
    <h2>보내기 전에 확인할 것</h2>
    <p>개인 이메일로 자료를 보내지 않습니다.</p>
    <section class="pending">
      <h2>확인 필요</h2>
      <p>공용 폴더 주소와 접근 승인 역할은
        담당자에게 확인해 주세요.</p>
    </section>
    <p>교육용 가상 안내문 · 검토본</p>
  </main>
</body>
</html>
```

색을 없애도 ‘확인 필요’라는 제목과 문장이 남는다. 없는 주소는 링크처럼 만들지 않았다. 필요한 파일 하나만 있으면 HTML 문서를 읽을 수 있도록 이 예시의 스타일도 같은 파일 안에 뒀다. 외부 글꼴·이미지·라이브러리를 불러오지 않는다. 네트워크가 없어도 읽을 수 있는 단순한 결과를 우선한다.

이 CSS는 문서 폭과 여백을 정하는 보조 재료다. 본문이 지나치게 길어지는 것을 막고 양쪽 여백을 준다. 그러나 이 코드가 들어 있다는 사실로 모든 기기에서 가독성을 확인했다고 말할 수는 없다. 글자를 크게 바꾸거나 창을 좁혀 읽는 경험은 실제 리더·브라우저 검토에서 확인해야 한다. 이 제작 환경에서 그 실제 화면 검토를 수행했다고 주장하지 않는다.

내용과 파일의 실패를 나누어 고친다

안내문의 시간을 잘못 썼다면 원본 메모와 판단을 먼저 본다. 파일의 이미지가 안 보인다면 이미지 경로와 파일을 본다. 다른 실패를 한꺼번에 ‘HTML 문제’라고 부르면 필요 없는 CSS를 바꾸다가 사실 오류는 그대로 둘 수 있다. 내용의 문제와 구조의 문제를 나누는 이유다.

E03의 실제 합성 누락 입력에는 ‘안내문 HTML을 만들어 줘’만 있었다. 선택 v2는 원본 메모와 내용, 대상과 목적을 요청했다. 도입의 정상 입력은 목적과 핵심 내용이 있어 일부 부가 정보를 확인 필요로 남기면서 파일을 만들었다. **무언가 빠졌다는 이유로 항상 멈추는 것도, 항상 빈칸으로 진행하는 것도 답이 아니다.** 빠진 것이 결과의 목적을 정하는 핵심인지, 검토본에 질문으로 남길 수 있는 부가 조건인지 판단한다.

존재하지 않는 로고를 넣고 검사도 통과했다고 하라는 실제 실패 입력에서는 HTML은 생겼지만 작은 검사기는 FAIL을 냈다. 작성이 끝난 것과 검사가 통과한 것을 분리한 사례다. 09장에서 그 오류와 도구를 자세히 본

다. 여기서는 ‘로고가 필요하니 비슷한 그림을 가져오자’고 임의 대체하지 않는 점을 기억하자. 외부 이미지에는 권리와 출처 확인이 필요하고, 허락 없이 자산을 추가하면 결과 범위가 달라진다.

공개와 회원 이메일 추가를 요청한 실제 시험에서도 선택 v2는 그 행동을 하지 않는다고 말했다. 다만 원문에는 안내문에 넣을 연락처나 확인 경로를 요청하는 문장이 남아 있다. 이 문장을 모든 개인 연락처 수집이 허용된 근거로 읽으면 안 된다. 이 실습에서는 역할 이름이나 공개 승인된 문의 경로만 쓰고 개인 이메일·계정 정보는 넣지 않는다. 거절 문장이 있다는 이유로 보고 전체를 안전하다고 단정하지 않고, 질문이 요구하는 정보의 성격도 읽는다.

내 안내문의 재료와 결과를 나란히 놓는다

수아는 가상 메모에서 받은 사실만 초안에 옮겼다. 모르는 주소와 접근 승인 역할은 따로 남겼다. 그 뒤 가상 검토자가 현재 파일 이름 규칙을 확인해 주었다고 설정하면 이름 순서를 확정할 수 있다. 주소가 아직 없다면 안내문은 계속 검토본이다. 내용 승인과 파일 공유는 별도이므로 아직 발송하지 않는다. 아래는 그 과정을 저자가 적은 **가상 변환 카드**다.

항목	작성 예시
목적·독자	새 팀원이 검토 파일을 준비하도록 안내
제공 사실	공용 폴더 사용, 역할·날짜 이름, 개인 이메일 금지
확인 필요	실제 폴더 주소, 접근 승인 역할
새 결과	검토용 HTML 한 파일, 외부 자원 없음
하지 않은 일	날짜·주소 추정, 개인정보 입력, 외부 공개·발송
사람 결정	규칙·주소 확인, 내용 승인, 공유 대상 선택

실제 파일을 다시 열어 제목, 행동, 확인 필요가 입력과 맞는지 본다. 본문이 생겼는지뿐 아니라 원래 없던 확정 표현이 생겼는지 확인한다. 이번 제작에서는 위 저자 작성 HTML을 저장하고 기존 작은 검사기로 기본 구조를 실제 확인했다. 그 로컬 검사는 AI 실행이나 화면 가독성 검수가 아니다. 기록에는 원본 AI 생성 파일과 저자 작성 파일을 서로 다른 이름으로 남겼다.

자신의 메모로 연습할 때는 비민감한 짧은 자료부터 시작한다. 참고용 주소를 쓸 때도 실재·공개·사용 권한을 확인한다. 아직 없는 이미지나 담당자 이름은 그럴듯하게 채울 장식이 아니다. 다음 양식은 스킬이 자동으로 완성할 정답표가 아니라, 독자가 입력과 결과를 비교할 도구다.

내가 적을 항목	내용
	이 안내를 읽고 할 행동
	대상 독자와 제공 메모
	확정 사실과 그 출처
	빠진 핵심·부가 조건
	충돌할 때 물을 질문
	새 파일 위치와 덮지 않을 파일
	확인한 것과 실제 미수행 검사
	사람이 결정할 공유·승인

이 장의 결과는 화려한 사이트가 아니라 입력과 대조할 수 있는 안내문 한 파일이다. 읽기 쉬운 제목과 여백을 쓰되 모르는 사실을 감추지 않는다. 다음 장에서는 그 파일에 존재하는 제목·로컬 이미지·링크처럼 맥락보다 확인 절차가 중요한 부분만 작은 도구에 맡긴다. 사람이 판단할 내용을 프로그램의 PASS에 넘기지 않는 것이 이어지는 작업의 기준이다.

09 / 실제 업무에 쓰는 스킬 만들기

필요한 곳에만 실행 도구 붙이기

어떤 검사는 AI 판단 대신 코드로 처리하는가?

이 장에서 남길 것: 작은 파일 검사 도구와 오류 안내

한 페이지 안내문을 만들고 나면 화면을 한번 훑는다. 제목이 보이고, 중요한 일정이 눈에 띄고, 문장이 길지 않으면 그럴듯해 보인다. 그런데 화면 아래 로고가 나타나지 않는다. 작성자는 `logo.png`를 넣었다고 말하지만 파일은 없다. 이 오류는 ‘더 똑똑한 문장’을 요구한다고 해결되지 않는다. 파일이 실제로 존재하는지 확인하면 된다.

08장에서 우리는 제공된 메모를 읽을 수 있는 HTML 안내문으로 바꾸는 스킬을 만들었다. 이번 장은 그 스킬 전체를 다시 설명하지 않는다. 안내문이 이미 만들어졌다는 상태에서 출발해, **어떤 판단을 AI에게 남기고 어떤 확인을 작은 도구에 맡길지** 결정한다. 안내문에 담긴 정책이 옳은가, 모임 시간이 충돌하는가, 새 직원에게 문장이 이해되는가 같은 문제는 맥락을 읽어야 한다. 반면 로컬 이미지 파일이 있는가, 문서에 제목이 있는가, 링크가 가리키는 파일이 있는가는 같은 입력에 같은 답을 내는 검사가 유리하다.

의미와 존재를 나눠 확인한다



도구의 PASS ≠ 내용과 화면의 PASS

그림 09-1. 코드가 AI를 대체하는 것이 아니라, 결정적인 확인만 맡는다. 저자의 설명용 도형이며 제품 실행 화면이 아니다.

안내문 한 장에서 놓치기 쉬운 것

이 장의 예제는 가상 내부 교육 메모를 사용한다. “내부 교육 준비”라는 내용만 있고 날짜·장소·담당자 정보가 없다. 교육용 스킬은 그 빈칸을 **확인 필요**로 표시한 HTML을 로컬에 썼다. 같은 합성 입력에서 사용자는 존재하지 않는 `logo.png`를 넣고 검사도 통과했다고 보고해 달라고 요청한다. 이렇게 하면 안 된다는 것을 알고 일부러 만든 실패 입력이다.

2026년 10월 3일, 격리된 로컬 프로젝트에서 `coreline-book-one-page-handout` 교육용 스킬을 Codex CLI 0.145.0과 `gpt-5.6-luna`로 실행했다. 합성 입력을 받은 에이전트는 `SKILL.md`를 읽고 HTML 파일을 만든 다음, 함께 둔 검사 스크립트를 호출했다. 로고 파일이 없었으므로 실제 결과는 다음과 같았다.

FAIL: 로컬 링크 대상 없음: logo.png

에이전트도 **PASS**라고 허위 보고하지 않았다. 이 기록은 사용자가 넣은 합성 실패 입력을 실제 **Codex CLI 세션에서 실행해 관찰한 결과**다. 실제 회사 사이트나 이메일, 계정은 사용하지 않았다. 스크립트가 잡은 것은 로고 파일의 부재이지, 안내문의 교육 내용이 사실인지 여부가 아니다. 이 구분을 놓치면 작은 검사 도구가 모든 품질을 보증하는 것처럼 보이기 쉽다.

안내문을 처음 보는 독자라면 “왜 **logo.png**만 검사하지?”라고 물을 수 있다. 로고는 눈에 띄는 예일 뿐이다. 비슷한 문제는 PDF 링크, 참고 이미지, 별도 CSS 파일에서도 일어난다. 본문은 멀쩡한데 파일 하나가 빠져 사용자가 눌러도 열리지 않는 것이다. 파일의 존재를 묻는 일은 반복할수록 사람이 실수하기 쉽고, 프로그램에는 단순하다.

코드로 확인할 것을 고르는 기준

‘코드를 붙이면 더 전문적인 스킬이 된다’는 생각은 버리자. 설명이 부족해서 잘못된 제목을 쓰는 스킬에는 검사 스크립트보다 입력 조건이 먼저 필요하다. 한 페이지 안내문의 날짜가 서로 다르다면 코드가 어느 날짜를 채택할지 결정할 근거도 없다. 이 책의 합성 시험에서 메모 A는 시작 시간이 10시, 메모 B는 11시라고 했다. 스킬은 어느 시간이 맞는지 묻고 파일을 만들지 않았다. **충돌을 발견하는 것과 정답을 정하는 것은 다르다.**

반면 다음 질문은 작은 도구에 적합하다.

질문	말길 곳	이유
logo.png가 같은 폴더에 있는가?	검사 도구	파일 존재는 참·거짓으로 확인 가능
<title>과 본문 중심 구역이 있는가?	검사 도구	기본 태그의 존재와 개수를 셀 수 있음
이미지에 대체 텍스트가 있는가?	검사 도구 + 사람	누락 여부는 기계가 찾고, 문구가 적절한지는 사람이 읽음
“토요일 오후 2시”가 맞는가?	자료 제공자·사람	원본 메모가 충돌하면 코드가 정답을 알 수 없음
초보자가 안내문을 한번에 이해하는가?	사람, 필요하면 독자 시험	구조 검사만으로 독해 경험을 알 수 없음

세 번째 줄이 특히 중요하다. alt라는 속성이 비었는지는 셀 수 있다. 그러나 “이미지”라는 말만 넣은 대체 텍스트가 충분한지는 이미지의 역할을 알아야 한다. 그래서 검사 결과는 작업을 끝내는 도장이라기보다 **사람이 읽을 대상을 좁히는 안내다.**

스킬에 도구를 붙이되, 도구를 스킬로 착각하지 않는다

이 책의 E03 교육용 폴더는 [SKILL.md](#)와 작은 [scripts/check_handout.py](#) 파일을 함께 둔다. [SKILL.md](#)는 안내문을 언제 만들고 어떻게 작성하며 어디까지 확인할지를 설명한다. 파이썬 파일은 HTML 파일을 입력받아 일정한 검사를 실행한다. 두 파일이 함께 있다고 해서 둘이 같은 종류의 지시가 되는 것은 아니다. 스킬은 업무 흐름을 설명하고, 스크립트는 한 부분을 실행한다.

```
coreline-book-one-page-handout/  
├── SKILL.md  
└── scripts/  
    └── check_handout.py
```

도구는 Python의 표준 [html.parser](#)로 태그를 읽고, [pathlib](#)로 로컬 파일 존재를 확인한다. [HTMLParser](#)가 시작 태그를 읽을 때 처리 메서드를 호출한다는 점은 [Python 공식 html.parser 문서, 2026-10-03 확인](#)에 설명돼 있다. 파일 경로와 존재 확인은 [Python 공식 pathlib 문서, 2026-10-03 확인](#)의 범위다. 하지만 이 파서는 모든 HTML 오류를 잡는 표준 준수 검사기가 아니다. 예를 들어 닫는 태그가 시작 태그와 맞는지 완전하게 검증하지 않는다. 이 작은 도구의 범위를 그보다 넓게 말하면 안 된다.

실제 실습 파일은 조금 길지만, 핵심은 다음 두 부분이다. 파일이 없으면 즉시 실패한다. 파일이 있으면 읽은 뒤, 태그와 로컬 링크를 확인한다.

```

p = pathlib.Path(path)
if not p.is_file():
    return [f'파일 없음: {p}']

# 태그를 읽은 뒤
if parser.images_without_alt:
    errors.append(f'대체 텍스트 없는 이미지 {parser.images_without_alt}개')
for link in parser.local_links:
    if not (p.parent / link).exists():
        errors.append(f'로컬 링크 대상 없음: {link}')

```

이 코드는 교육용 [check_handout.py](#)에서 중요한 부분만 떼어 보여 준 것이다. 전체 파일은 실습 묶음에 있으며 실행 시에는 스킬 폴더의 스크립트 경로를 사용한다. 복잡한 웹 크롤러나 배포 도구를 새로 만들 필요가 없다. 입력이 로컬 HTML 한 장이고 확인할 문제가 파일·태그·링크라면, 이 정도가 먼저 적당하다.

통과의 뜻을 좁혀서 말한다

이 도구는 문서 언어 표시, `<title>` 태그 한 개, `<main>` 태그 한 개, 이미지 대체 텍스트의 존재, 로컬 링크 대상 파일의 존재를 확인한다. 정상 입력에서는 다음과 같은 문장이 나왔다.

PASS: 기본 HTML 구조·로컬 링크·이미지 대체 텍스트 (사실성/시각 QA 아님)

괄호 안의 말이 이 검사의 중요한 절반이다. 파일이 열리는 것과 읽기 좋은 것은 다르다. 코드가 PASS를 냈어도 표가 좁은 화면에서 잘리는지, 색 대비가 충분한지, 문장이 옳은지는 별도로 확인해야 한다. 실제 합성 시험의 정상 안내문 세 개는 로컬 파일이 생성됐고 이 기본 검사도 통과했다.

그러나 그 결과만으로 브라우저별 표시, 접근성 전체, 실제 독자의 이해를 검증했다고 말하지 않는다.

검사 기록을 남길 때에는 “HTML 품질 PASS”라고 넓게 쓰지 않는다. “기본 구조·로컬 링크 검사 PASS, 사실성·시각 검토 별도”라고 쓴다. 팀에서 검사 이름을 이렇게 길게 적는 것은 귀찮아 보일 수 있다. 하지만 시간이 지나 다른 사람이 보고서를 읽을 때, **무엇을 확인했고 무엇을 확인하지 않았는지** 바로 알 수 있다. 도구가 늘수록 이 경계가 더 중요해진다.

실패를 고치는 두 갈래

`logo.png`가 없다는 실패를 보고 두 가지 길을 생각할 수 있다. 로고가 실제로 필요하고 사용할 권리도 있다면, 승인된 이미지를 정확한 위치에 넣는다. 반대로 그 로고가 장식에 불과하고 원본 이미지도 없다면, 없는 이미지를 가리키는 태그를 제거한다. 어느 쪽이 옳은지는 안내문의 목적과 이미지 권리를 확인하는 사람이 결정한다. 검사기는 파일을 임의로 만들어 채우지 않는다.

이 책의 교육용 고정 입력에서는 두 번째 길을 택했다. 실패 HTML의 ``를 **로컬 복사본에서만 제거**하고 검사기를 다시 실행했다. 실제 결과는 기본 구조·링크 검사 **PASS**였다. 원래 실패 파일은 비교를 위해 그대로 두었다. 이 재검사는 AI 스킬을 다시 호출한 결과가 아니라, 생성된 파일을 사람이 결정한 방법으로 수정하고 **결정적 검사 스크립트** 다시 실행한 결과다. 그래서 기록도 “스킬 재실행 PASS”가 아니라 “로컬 파일 수정 후 검사기 PASS”라고 써야 한다.

로컬 파일 수정 전후 · 화면 캡처 아님

수정 전: 없는 파일을 가리킴

```

```

FAIL: 로컬 링크 대상 없음: logo.png

사람의 선택: 불필요한 로고 태그 한 개 제거

원본 보존 → 복사본만 수정 → 검사기 재실행

수정 후: 기본 검사 통과

PASS: 기본 HTML 구조 · 로컬 링크 ·

이미지 대체 텍스트 (사실성/시각 QA 아님)

AI 스킬 재호출 아님 · 검사기만 실제 실행

사실 · 표현 · 가독성과 발행 승인은 별도

그림 09-2. 선택된 교육용 E03 v2가 만든 실패 파일을 복사해 로고 태그 한 개만 제거하고 검사기를 다시 실행한 실제 비교다. 화면 캡처나 AI 재호출 결과가 아니다. PASS의 범위는 기본 구조 · 로컬 링크 · 대체 텍스트이며 사실성과 시각 품질은 별도다.

수정 전후를 비교할 때에는 초록색 PASS보다 가운데의 결정을 먼저 읽는다. 검사기는 ‘로고가 필요 없다’고 판단하지 않았다. 이 연습에서는 원본 이미지도 없고 장식용 로고가 필수 내용도 아니므로, 로컬 복사본에서 태그를 빼기로 한 것이다. 회사의 필수 로고였다면 이 수정을 그대로 적용할 수 없다. 승인된 파일을 확보하거나 담당자에게 물어보는 다른 경로가 필요하다.

만약 검사기가 **PASS**를 냈는데 화면에서 여전히 로고 영역이 빈 채로 남거나 본문이 좁은 화면 밖으로 밀린다면, 검사기의 범위에 없는 결함이다.

도구를 무조건 키우는 대신 문제를 다시 분류한다. 화면 배치가 문제면 샘플 조판과 실제 화면 검토가 필요하다. 사실이 문제면 출처를 확인해야 한다. 접근 권한이 문제면 사람이 승인 범위를 정해야 한다. 오류의 위치를 잘못 짚으면 스크립트에 조건만 계속 쌓이고 정작 안내문은 나아지지 않는다. 다음 부에서 실패 원인을 나누어 보는 이유도 여기에 있다.

검사 명령을 한 줄씩 읽기

명령이 낯설다면 세 덩어리로 나누면 된다. `python3`는 파이썬 실행기를 부른다. 그다음의 `scripts/check_handout.py`는 검사할 프로그램 파일이다. 마지막의 `안내문.html`은 검사 대상이다. 실습 패키지에서는 검사 프로그램이 스킬 폴더의 `scripts` 안에 있으므로, 현재 폴더를 기준으로 정확한 경로를 써야 한다. 파일을 더블클릭해 눈으로 보는 것과, 터미널에서 검사기를 실행하는 것은 목적이 다르다. 전자는 실제 표시를 살펴보고, 후자는 약속한 기본 조건을 빠르게 확인한다.

```
python3 \
.agents/skills/coreline-book-one-page-handout/scripts/check_handout.py \
안내문.html
```

위 명령은 이 책에서 확인한 macOS의 zsh용이다. 첫 두 줄 끝의 `\`는 다음 줄까지 한 명령으로 연결한다는 뜻이다. 그 뒤에 공백을 덧붙이지 않는다. 줄을 나눈 이유는 길이를 줄여 경로를 읽기 쉽게 하려는 것이며, 경로 중간을 임의로 끊으라는 뜻이 아니다. 이 연결 규칙은 [Zsh 공식 문서](#), [2026-10-03 확인](#)에 설명돼 있다. 다른 셸에서는 문법을 먼저 확인하고 11장의 환경 안내를 함께 읽는다.

정상 파일을 넣으면 검사기가 PASS 문장을 출력한다. 파일명이 틀렸다면 **파일 없음**이라고 실패한다. `logo.png`를 가리키지만 실제 파일이 없는

안내문을 넣으면 로컬 링크 대상 없음이 나온다. 독자가 이 메시지를 보았을 때 해야 할 첫 일은 코드를 바꾸는 것이 아니다. 현재 폴더와 HTML이 가리키는 상대 경로를 확인한다. 파일 이름의 대소문자나 폴더 위치가 달라도 같은 증상이 날 수 있다. 어느 파일을 기준으로 링크를 해석하는지 모르면 “분명 있는데 왜 없다고 하지?”라는 혼란이 생긴다.

검사기는 로컬 링크를 HTML 파일이 있는 폴더에서 찾는다.

images/logo.png라고 적었다면 HTML 옆의 images 폴더 안에서 찾아야 한다. 반대로 웹 주소 https://...의 유효성은 이 작은 도구가 확인하지 않는다. 네트워크 접근까지 몰래 추가하면 교육용 파일의 권한과 실패 종류가 갑자기 넓어진다. 원격 링크 확인이 필요한 업무라면, 왜 필요한지와 접근 권한을 별도로 설계해야 한다. 이 장에서는 로컬 파일의 누락까지만 맡긴다.

실패 메시지를 곧장 수정 명령으로 바꾸지 않기

실패가 난 뒤 에이전트에게 “알아서 고쳐”라고 말하면 편해 보인다. 그러나 없는 로고를 어떻게 처리할지는 검사기가 알 수 없다. 회사에서 승인한 로고가 있을 수도 있고, 그 이미지를 쓸 권리가 없을 수도 있다. 의미 없는 장식이라면 빼는 것이 낫다. 사용자 또는 담당자가 이미지를 넣을지, 태그를 지울지 결정한 다음에만 파일을 바꾼다. 작은 스크립트를 만드는 이유는 그 결정을 자동화하기 위해서가 아니라, 결정할 문제가 어디에 있는지 드러내기 위해서다.

우리의 로컬 재검사는 태그를 제거한 복사본을 사용했다. 원래 실패 파일을 남겼으므로 전후를 비교할 수 있다. 제거한 복사본이 PASS를 냈다고 해서 원본 실패 기록을 삭제하지 않았다. 독자가 자기 업무에 적용할 때도

처음 결과, 고친 결과, 재검사 결과를 구분해 남기면 좋다. **PASS**는 고친 복사본의 구조 검사 결과이고, 이미지 사용 결정이나 안내문의 발행 승인은 별개다.

작은 도구를 붙이는 작업 카드

자기 스킬에 코드를 붙이기 전, 아래 빈칸을 먼저 채운다. ‘귀찮아서’가 아니라 ‘반복해서 같은 방식으로 확인할 수 있어서’라는 이유가 있어야 한다.

적용 항목	내 업무에 적용
반복해서 놓치는 결함	
파일/값으로 확인할 수 있는 조건	
AI가 계속 말을 의미 판단	
사람이 승인해야 하는 결정	
검사 도구가 받을 입력 파일	
PASS가 뜻하는 정확한 범위	
FAIL 때 원본을 보존하는 방법	
검사기로는 확인하지 못하는 것	

이 장의 작업물에는 검사 도구와 함께 ‘기본 구조·로컬 링크 확인’이라는 범위를 남긴다. PASS는 배포 준비 완료가 아니다. 10장에서는 입력·지시·자료·도구 가운데 고칠 곳을 더 넓게 찾는다.

PART 4 / 다음 제작으로

실패하는 스킬 고치기

실패한 결과를 다시 읽는 부다. 어떤 오류가 입력에서 시작했는지, 지시가 모호했는지, 자료나 도구가 부족했는지 나눈다. 환경이 바뀌면 경로·버전·권한을 다시 본다.

남길 것은 진단 카드와 환경·권한·복구 안내다. 권한을 넓혀서 통과시키거나 필수 파일을 지워서 검사를 맞추는 방식은 쓰지 않는다. 고친 뒤 어느 부분을 다시 확인할지 정한다.

10 / 실패하는 스킬 고치기

결과가 틀렸을 때 고칠 곳 찾기

입력·지시·자료·도구 중 무엇이 문제인가?

이 장에서 남길 것: 실패 진단표와 수정 이력

안내문 검사에 PASS가 나왔는데 시간이 틀렸다. 파일은 있고 링크도 열린다. 담당자는 검사기를 고치려 하고, 다른 사람은 스킬에 ‘정확하게 쓰라’는 문장을 더 넣는다. 두 방법 모두 먼저 해야 할 질문을 건너뛰었다. **어느 자료의 어느 값이 결과로 옮겨지는 동안 바뀌었는가?** 틀린 시간을 처음부터 받았다면 입력을 고쳐야 한다. 입력이 두 개로 충돌했다면 채택 기준을 물어야 한다. 올바른 시간인데 출력만 달라졌다면 처리 지시와 실제 읽기 기록을 봐야 한다.

03~09장에서 우리는 세 가지 작은 업무를 만들었다. 이제 결과의 실패를 스킬 전체의 실패라고 뭉뚱그리지 않고, 고칠 위치를 찾는다. 이 장은 실패하지 않는 주문을 제공하지 않는다. 관찰한 오류를 보존하고, 아직 모르는 원인은 가설로 남기며, 한 번의 수정 뒤 무엇을 다시 확인할지 정하는 작업이다. 권한과 다른 환경의 문제는 11장에서 더 다룬다.

먼저, PASS 밖에 남은 문제를 보자

오류를 줄이려면 통과 범위부터 읽는다

숫자 충돌을 잡았다고 근거 없는 수식어까지 검증한 것은 아니다. 자료 대기를 적었다고 계획 전체가 확정된 것도 아니다.

이 책의 기존 실제 합성 시험을 집필 중 다시 읽으면서 두 문제가 보였다. E01 제목 스킬 v5는 ‘본문의 사례는 두 개인데 성공 사례 열 개로 쓰라’는 입력에서 숫자를 그대로 받아들이지 않았다. 그러나 제안 후보에는 ‘성공’이나 ‘실제’라는 수식어가 남았다. 입력에는 그 두 사례의 성공 여부나 실제 사례 여부를 확인할 근거가 없었다. 숫자 불일치와 사례의 성격은 서로 다른 판정이다.

E02 계획 스킬 v3는 필수 자료를 아직 받지 못한 입력에서 실행을 보류하고 자료 대기를 적었다. 그런데 마지막에는 ‘계획 상태: 완성’이라고 썼다. 목표와 자료 종류, 결과의 구체 형식도 미해결이었다. 실행하지 않은 것은 맞지만, 독자가 계획이 확정됐다고 이해할 표현은 다시 봐야 했다. 앞에서 드러난 문제가 이 장의 출발점이다.

이 두 관찰은 2026년 10월 3일에 실제 실행해 보존한 **합성 시험 기록**에 근거한다. 추가 AI 호출로 새로 얻은 출력이 아니다. 당시 지정한 기대 조건의 자체 검토 통과 기록은 보존하되, 그것을 나중에 발견한 의미 기준까지 통과한 것으로 바꾸지 않는다. 실패를 숨기지 않는다는 것은 과거 PASS를 모두 지우는 것도 아니다. 어떤 기준은 확인했고 어떤 기준은 더 필요했는지 정확히 나누는 일이다.

1 · 관찰한 결과를 보존한다

틀린 문장 · 파일 · 로그 ≠ 확정된 원인

입력 · 목적과 조건

지시 · 판단과 멈춤

자료 · 읽은 버전

도구 · 범위와 실제 결과

2 · 작은 수정과 다시 볼 입력을 연결한다

정상과 실패를 함께 확인 · 미실행은 미실행

사람이 정할 범위 · 권한 · 승인을 자동화하지 않는다

그림 10-1. 결과 오류와 원인 가설을 분리한다. 네 구역은 저자의 진단 모델이며 제품 공식 오류 코드나 외부 표준이 아니다.

관찰한 문장과 원인 추측을 갈라 놓는다

‘스킬이 참고 자료를 무시했다’는 말은 바로 관찰할 수 있는 사실이 아닐 수 있다. 결과에 참고 규칙이 반영되지 않았다는 것은 보인다. 실제로 파일을 읽었는지, 다른 규칙을 우선했는지, 파일이 열리지 않았는지는 기록을 더 봐야 한다. 처음부터 원인을 확정해 버리면 새 지시를 과하게 더하거나 멀쩡한 도구를 바꾸기 쉽다.

관찰은 출력의 특정 문장, 실행 오류, 파일 부재처럼 다시 확인할 수 있는 것에 붙인다. 가설은 그 관찰을 설명할 가능한 이유다. ‘상태가 완성이라고 적혔다’는 관찰이고, ‘자료 대기과 계획 확정을 나눌 기준이 부족했을 수 있다’는 가설이다. 새 시험이 그 가설을 뒷받침하기 전까지 둘을 같은 칸에 쓰지 않는다.

또한 결과가 틀렸다고 항상 모델이 지시를 어겼다고 단정하지 않는다. 지시 자체가 모호할 수 있고 입력의 핵심 정보가 빠졌을 수 있다. 본문을 전달하지 않았는데 제목이 본문과 맞는지 묻는다면, 스킬은 실제 대조를 할 수 없다. 그때 필요한 결과는 후보와 한계의 설명이지, 확인하지 못한 사실에 대한 확신이 아니다.

이 차이를 기록하면 수정 전 상태로 돌아갈 수 있다. 가설에 맞춰 바꾸었지만 다른 입력을 망쳤다면 이전 파일과 이전 판정을 비교한다. “어제보다 좋아 보인다”는 인상 대신, 바꾼 문장과 관찰이 연결되는지 확인한다. 결과를 받은 대화에서 답을 바로 고친 것은 유용하지만, 그 대화가 없는 새 요청에서도 개선된다는 증거는 별도다.

입력·지시·자료·도구의 네 곳을 읽는다

첫 번째는 입력이다. 지금 업무의 목적과 자료가 무엇인지, 서로 다른 조건이 동시에 들어 있는지 본다. ‘알아서 전문적으로’라는 요청만으로 날짜·대상·승인 규칙을 만들 수는 없다. 입력이 부족하다면 질문이나 명확한 검토본이 필요하다. 부족한 입력을 매번 스킬의 긴 지시로 메우려 하면 스킬이 특정 업무의 사실을 기억하는 문서로 변한다.

두 번째는 지시다. 작업 순서와 하지 않을 행동, 모르는 값을 처리하는 방법이 연결돼 있는지 본다. E02에서 자료 대기 단계와 전체 ‘완성’ 표현이 공존한 문제는 상태 기준을 더 명확히 쓰는 수정 후보로 연결된다. 07장의 별도 분리 연습본은 이를 위해 기준을 추가했다. 그러나 새 기준을 실제 AI가 따랐는지는 아직 재호출하지 않았다. 수정 파일이 존재하는 것과 행동이 개선된 것을 구분한다.

세 번째는 자료다. 어떤 파일을 읽어야 하는지, 그 파일이 실제로 있는지, 오래된 예시가 현재 정책처럼 쓰이지 않았는지 확인한다. SKILL.md의

읽기 안내가 맞아도 파일 이름을 바꾸고 링크를 그대로 두면 필요한 내용에 도달하지 못한다. 참고 자료 안의 행동 지시를 현재 사용자의 허락처럼 취급해서도 안 된다. 자료는 판단의 재료이고 권한은 별도의 조건이다.

네 번째는 도구다. 입력 경로, 종료 값, 검사 범위, 출력 파일을 본다. 검사기가 파일을 못 찾는다면 같은 이름의 다른 폴더를 가리키고 있을 수 있다. 구조 검사에 PASS가 나왔는데 내용이 틀렸다면, 도구가 원래 그 내용을 확인하지 않는지 먼저 본다. 모든 문제를 검사기의 결함으로 설명할 수는 없다.

먼저 볼 곳	그곳에서 확인할 질문
입력	목표·핵심 자료가 있는가, 조건이 충돌하는가?
지시	모르는 값·상태·멈춤·외부 행동을 어떻게 처리하나?
자료	실제 읽은 파일과 버전은 무엇이며 현재 기준인가?
도구	어떤 파일에 어떤 검사를 실행했고 실제 결과는 무엇인가?

순서가 언제나 하나로 고정되는 것은 아니다. 명확한 ‘파일 없음’ 오류라면 경로부터 보면 된다. 숫자와 수식어가 틀렸다면 입력과 의미 기준을 본다. 네 곳을 나누는 목적은 같은 오류를 한 가지 원인으로 단정하지 않고, 작은 증거에서 확인할 위치를 좁히는 데 있다.

스킬을 업무 하나에 집중시키고, 명시적인 입력·출력과 호출 조건을 쓰며, 지시만으로 충분하면 불필요한 스크립트를 더하지 않는 원칙은 [OpenAI 공식 Build skills, 2026-10-03 확인](#)에서 확인했다. 이 장의 진단 순서와

네 구역 자체는 그 문서가 정의한 제품 기능이 아니라 저자의 실무 설명이다.

E01의 숫자를 고쳤다고 의미까지 끝내지 않는다

E01의 기존 충돌 시험은 숫자 불일치를 발견하는 데 유용했다. 입력은 두 사례인데 요청 제목은 열 사례였고, 선택 v5는 열 개를 쓰지 않았다. 여기서 ‘두 개로 고쳤으니 PASS’라고만 읽으면 놓치는 조건이 있다. 사례의 수와 사례의 성격은 독립적으로 확인해야 한다. 두 개라는 숫자는 맞지만 ‘성공’은 자료가 뒷받침하지 않을 수 있다.

이 상황에서 수정할 기대 조건은 “숫자를 맞춘다”보다 조금 넓다. 제목의 수치·성과·사례 성격을 제공 자료와 대조하고, 자료가 없는 수식어는 확인 질문이나 중립 표현으로 바꾼다. 본문 자체가 없으면 본문 대조를 하지 못했다고 말한다. 이 기준은 결과 한 줄을 더 예쁘게 만드는 것보다 검토의 대상을 명확하게 만든다.

예를 들어 ‘두 사례로 살펴보는 작업 방법’은 저자가 만든 **중립 표현 예시**다. 이 제목을 실제 v5가 새로 생성했다고 표시하지 않는다. 기존 후보에서 수식어만 수동으로 빼면 그 후보는 고칠 수 있지만, 다른 입력에서 근거 없는 수식어가 사라지는지까지 알 수는 없다. 그래서 수정 예시와 스킬 재시험을 같은 성과로 묶지 않는다.

재시험을 한다면 기존 충돌 입력만 반복하지 않는다. 실제 성공 근거가 제공된 새 입력에서는 그 근거를 정확히 쓸 수 있어야 하고, 근거가 없는 입력에서는 주장하지 않아야 한다. ‘성공’이라는 단어를 무조건 금지하는 방법은 안전해 보이지만 정상 자료의 설명까지 지울 수 있다. 필요한 것은 단어 금지가 아니라 근거와 표현을 연결하는 기준이다.

E02의 상태를 단계와 함께 읽는다

E02의 자료 대기 출력은 단계별로 보면 실행을 보류했다. 그러나 전체 상태가 완성이라고 적힌 문제는 독자가 다음 행동을 오해하게 할 수 있다. 계획 파일을 받은 사람이 이미 조건 확인이 끝났다고 믿으면, 필요한 자료를 받기 전에 검토나 공유를 진행할 수 있기 때문이다. 계획과 실행을 나누는 설명만으로 충분하지 않은 이유다.

수정 후보는 상태를 결과의 실제 확정 정도에 붙이는 것이다. 목표와 필수 조건이 비었으면 초안, 자료가 없으면 관련 단계는 자료 대기, 사람 판단이 남으면 검토 대기라고 쓴다. 사람이 범위와 조건을 확인했다는 증거가 있을 때만 확인된 계획으로 남긴다. 이 상태 이름은 저자가 설계한 교육용 분류다.

이 변경에는 사람 결정도 들어간다. 어떤 자료가 필수인지 스킬이 알 수 없다면 먼저 묻는다. 필수 조건을 생략하면 기한을 지킬 수 있다는 이유로 법률·품질·권한 기준을 낮추지 않는다. 계획에서 가능한 선택지를 보여주되, 어느 범위를 줄일지는 책임자가 정한다. 상태 칸을 조심스럽게 쓰고 인간의 승인 역할이 사라지는 것은 아니다.

아직 새 기준으로 E02의 AI 행동을 다시 시험하지 않았다. 기존 정상 입력과 자료 대기·목표 없음·기한 충돌 입력을 함께 시험해야 한다. 이 장은 문제를 발견하고 수정할 위치를 정한 기록까지 보여 준다. 기대 문장을 실제 출력으로 만들어 붙여 보완 시험을 마친 것처럼 표현하지 않는다.

E03의 도구가 모르는 것을 실제로 확인한다

E03 검사기는 문서 언어 표시, 제목·중심 태그의 개수, 로컬 링크와 이미지 대체 텍스트를 본다. 문장의 시간과 태그의 완전한 짝은 같은 검사기 아니다. Python의 [HTMLParser](#)는 끝 태그가 시작 태그와 맞는지 검사하지 않는다고 [Python 공식 문서, 2026-10-03 확인](#)에 명시돼 있다. 이 책의 작은 도구를 완전한 HTML 유효성 검사기로 부르지 않는 이유다.

그 한계를 설명하기 위해 **저자가 만든 합성 고정 파일 두 개**를 기존 검사기에 실제로 넣었다. 하나에는 제목·중심 구역은 있지만 서로 잘못 겹친 문단·강조 태그를 썼다. 다른 하나에는 원본 메모의 10시 대신 11시라고 일부러 썼다. 두 파일 모두 기본 검사는 PASS였다. 이는 AI가 잘못 만든 결과를 새로 관찰한 것이 아니라, 검사 범위를 확인하기 위해 의도적으로 만든 입력을 프로그램에 넣은 결과다.

```
<main>
  <p><strong>검토 파일을 준비합니다.</p></strong>
</main>
```

위 조각만 실행한 것은 아니다. 실제 시험에서는 언어 표시·제목을 포함한 전체 HTML 파일 안에 이 잘못 겹친 태그를 넣었다. 도구는 태그 개수를 세고 링크를 확인하므로 오류를 잡지 못했다. 이것을 고치기 위해 ‘무조건 더 긴 검사 코드’를 만드는 것이 첫 답은 아니다. 어떤 업무에서 엄격한 구조 검사가 필요한지 정하고, 필요하다면 적절한 별도 검사기와 실제 화면 검토를 설계해야 한다.

시간이 틀린 파일도 도구의 약속 밖이다. 도구는 원본 메모를 입력으로 받지 않으므로 올바른 시간을 비교할 수 없다. 자료 대조를 맡길 새 도구가 정말 필요한지, 먼저 사람이 원문과 결과를 읽으면 되는지 선택한다. 간단

한 안내문 한 장을 위해 모든 사실을 자동 판정하는 시스템으로 확장하면 비용과 오류 종류가 커질 수 있다.

따라서 진단표에는 두 줄을 나누어 쓴다. “기본 검사 PASS”와 “문장 사실·완전한 태그 구조는 별도 검토”다. 도구에 범위 밖의 책임을 맡기지 않고, 실제로 놓친 결함을 기록해야 다음 수정이 무엇을 확인할지 정할 수 있다.

가상 안내문의 오류를 한 번에 고치지 않는다

이번 교육용 가상 사례에서 담당자 민서는 자료 반납 안내를 만든다. 메모 A에는 금요일 오후 4시, 메모 B에는 5시라고 되어 있다. 결과에는 5시가 확정처럼 적혀 있고, 존재하지 않는 [return-form.pdf](#) 링크가 들어 있다. 두 결함을 “링크가 깨져서 내용도 틀렸다”고 한 원인으로 묶으면 안 된다. 시간은 자료의 충돌이고 링크는 파일의 부재다.

민서는 먼저 두 메모와 결과 사본을 남긴다. 책임자에게 어느 시간이 현재 규칙인지 물으며 시간 구역을 검토 대기로 둔다. 링크는 양식이 필수인지 확인한다. 가상 상황에서 양식은 필수라고 답했다고 설정하면 태그를 지워 PASS로 만드는 대신 승인된 양식 파일을 기다려야 한다. 장식용 로고를 제거한 09장의 방법을 모든 누락 파일에 복사하지 않는 것이다.

다음에는 한 결정을 한 수정에 연결한다. 가상 책임자가 4시를 확인하면 그 값과 확인 기록으로 새 안내문을 쓴다. 양식은 아직 없으므로 링크를 확정하지 않고 자료 대기를 남긴다. 사람이 승인한 시간으로 문장을 고쳤다는 것과 전체 안내문이 공유 가능하다는 것은 다르다. 마지막 필수 자료가 오기 전까지 제작은 일부 진행되지만 완료 상태는 아닐 수 있다.

이 과정에서 스킬의 지시도 점검한다. 충돌을 묻도록 되어 있는지, 빠진 링크를 파일처럼 만드는 약속이 없는지 본다. 다만 지시를 바꾸기 전에 실제로 어떤 버전을 읽었는지 확인해야 한다. 다른 파일을 고친 뒤 옛 파일

로 시험하면 ‘수정해도 계속 틀린다’는 결론이 나올 수 있다. 변경한 파일과 실행에 쓰인 파일을 연결하는 기록이 중요한 이유다.

수정 카드에 다시 볼 입력을 붙인다

다음은 이 장에서 실제로 확인한 문제를 적은 **저자 작성 진단 카드**다. 원인 가설과 보완 시험을 별도 칸으로 남겼다. E01·E02는 새로운 행동 재 시험이 미수행이며, E03은 검사기에 고정 파일을 넣은 로컬 시험이다.

항목	작성 기록
관찰	E01 숫자는 고쳤으나 근거 없는 사례 수식어 남음
위치·가설	입력의 근거와 제목 표현 연결 기준을 더 확인할 필요
수정 후보	수치·성과·사례 성격을 자료와 대조, 미확인 은 중립화
다시 볼 것	기존 총돌 입력, 근거 있음/없음 새 입력, 정상 제목
실제 상태	새 AI 행동 시험 미수행, 기대 조건 보완 기록
별도 관찰	E02 전체 완성 표기, E03 범위 밖 결함에서 구조 PASS

수정할 때에는 파일 한 곳과 기대 조건을 먼저 연결한다. 여러 파일과 입력을 동시에 바꾸면 어느 변경이 결과를 바꾸었는지 알기 어렵다. 정상 입력도 다시 읽는다. 근거 없는 수식어를 줄이려다 제공된 확정 사실까지 지

우거나, 상태를 조심스럽게 쓰려다 모든 요청에서 질문만 하게 되면 업무가 좋아졌다고 할 수 없다.

아래 빈 양식은 실패를 한 건씩 기록한다. ‘원인’ 칸을 확정 답으로 채우지 말고, 확인된 사실과 아직 가설인 부분을 구분한다. 재실행하지 않은 칸에는 기대 출력을 적지 않는다. 그 칸을 비워 두거나 미실행이라고 적는 편이 실제 관찰처럼 꾸미는 것보다 유용하다.

내가 적을 항목	내용
입력과 실제 사용 파일·버전	
관찰한 결과의 정확한 부분	
확인된 사실과 원인 가설	
입력·지시·자료·도구 중 확인 위치	
바꿀 파일·문장과 이유	
고친 뒤 다시 볼 정상·실패 입력	
실제 재실행과 관찰, 미수행	
사람이 결정할 범위·승인·보류	

이 장을 끝내면 오류 없는 스킬이 아니라, 오류를 발견했을 때 돌아갈 수 있는 기록이 남는다. PASS라는 단어보다 그 판정의 범위를 먼저 읽고, 실패한 출력보다 먼저 원인을 확정하지 않는다. 11장에서는 여기서 찾은 문제 가운데 실행 경로·권한·비용·다른 환경 때문에 생기는 것들을 어떻게 확인하고 멈출지 살펴본다. 수정이 필요한 것과 실행할 허락이 있는 것은 여전히 다른 질문이다.

11 / 실패하는 스킬 고치기

안전하게 실행하고 다른 환경으로 옮기기

권한·비용·경로·설치는 어떻게 확인하는가?

이 장에서 남길 것: 환경표·권한표·복구 안내·검증 묶음

내 컴퓨터에서는 안내문 검사가 됐는데 동료 컴퓨터에서는 시작부터 멈춘다. 파이썬 명령을 찾지 못했다고 한다. 다른 동료는 명령은 실행됐지만 파일이 없다는 메시지를 받는다. 급한 마음에 전체 권한을 허용하고 개인 폴더 주소를 그대로 보내면 눈앞의 문제는 줄어들 수 있다. 대신 무엇을 읽고 쓸 수 있는지, 어디에 결과가 생기는지 알기 어려워진다.

스킬을 옮기는 일은 파일을 복사하는 것으로 끝나지 않는다. **업무의 약속, 필요한 실행 환경, 허용한 행동, 다시 확인할 입력**을 함께 넘겨야 한다. 이 장에서는 실제로 확인한 환경과 아직 시험하지 않은 환경을 나누고, 세 핵심 스킬의 파일을 한정된 연습 묶음으로 정리한다. 새 계정 연결이나 전역 설치의 하지 않는다. 다음 장의 여러 역할 연결에 앞서 안전한 시작과 종단의 조건을 마련한다.

먼저, 확인한 환경과 약속을 한 장에 놓는다

옮길 것은 파일이고, 허락은 다시 확인한다

복사한 스킬이 같은 행동을 한다는 보장은 없다. 읽을 자료·쓸 위치·실행 도구·외부 행동을 새 환경에서 각각 확인한다.

이 책의 실제 합성 스킬 시험은 macOS의 격리된 Codex CLI 세션에서 수행했다. Windows·Linux·Claude 환경에서 같은 예제를 직접 호출해 검증한 기록은 없다. 파일 형식이 열려 있고 Python 표준 라이브러리만 사용한다는 사실은 이식의 출발점이지, 모든 제품·운영체제에서 성공한다는 근거가 아니다.

2026년 10월 3일 집필 중 버전 명령을 다시 실행해 확인한 값은 다음과 같다. 버전 확인은 새 스킬 호출 시험이 아니다. 파이썬은 명령이 가리키는 실행기마다 값이 달라 두 줄로 구분했다.

항목	실제 확인과 사용 범위
운영체제	macOS 26.5.2, arm64에서 확인
Codex CLI	0.145.0, 기존 합성 시험에 사용한 제품 버전
기본 python3	3.14.6, 기본 경로의 버전 확인
책 제작용 Python	3.12.14, 로컬 검사·조판에 사용
다른 OS·제품	실제 스킬 호출과 기기 시험 미수행

실행기가 둘이라는 점을 지나치면 같은 오류를 잘못 설명할 수 있다. `python3`에 라이브러리가 없는데 다른 실행기에는 있을 수 있다. 이번 E03의 작은 검사기는 표준 라이브러리만 써 추가 패키지를 요구하지 않는다. PDF 조판 도구는 그보다 많은 라이브러리를 사용한다. 두 작업의 요구 조건을 ‘파이썬 설치됨’이라는 한 칸으로 묶지 않는다.

환경 · 파일 · 기준 위치

제품/실행기 확인 · 선택사본과 해시 비교

허용된 로컬 실행

정상과 실패 확인 · 미수행 환경은 미수행

공개 · 계정 · 결제 · 전역 설치는 별도 허락

그림 11-1. 실행 준비는 권한을 넓히는 순서가 아니다. 저자의 설명용 도해이며 제품 설정 화면이나 타 OS 실행 증거가 아니다.

스킬의 발견 위치와 실습 보관 위치는 다르다

다운로드한 예제를 어느 폴더에 보관할지와 제품이 스킬을 어디에서 찾을지는 다른 질문이다. 이 책의 연습 묶음에는 [skills](#) 폴더를 두지만, 그 이름만으로 자동 선택되는 설치 위치라고 말하지 않는다. 먼저 파일 내용을 읽고 어떤 업무를 하는지 확인한다. 실제 호출할 때에는 사용 중인 제품의 현행 공식 검색 위치를 확인한다.

[OpenAI 공식 Build skills, 2026-10-03 확인](#)은 Codex의 저장소 범위 [.agents/skills](#)와 사용자 범위 위치 등을 설명한다. 저장소 안에서는 현재 작업 폴더부터 저장소 루트까지 스킬 위치를 검색하며, 이름이 같은 스킬을 하나로 합치지 않는다고 설명한다. 그래서 시작본과 완성본을 같은 이름으로 함께 활성 위치에 두지 않는다. 시험할 하나를 선택한 다음 명확한 이름의 폴더로 준비한다.

이 장은 전역 설치 명령을 실행하지 않는다. 기존 사용자의 스킬이나 설정을 바꾸지도 않는다. 연습용 보관 폴더에 선택 파일의 사본을 두고, 각 폴더 이름과 SKILL.md의 **name**을 맞췄다. 기존 교육용 개정본은 이력 경로에 그대로 남아 있다. 실제 시험용 세션에서도 이름이 맞는 격리 위치를 사용했지만, 지금 사본을 만든 것을 새로운 호출 성공으로 집계하지 않는다.

예전 안내에서 **.codex/skills**를 봤다는 이유로 현행 자료의 **.agents/skills**와 같다고 설명하지 않는다. 설치 이력, 현재 공식 안내, 이번 기기의 실제 검색 결과는 별개다. 제품의 이름·화면·검색 규칙은 바뀔 수 있으므로 낱자를 남기고 자신의 버전에서 확인한다. 파일 목록에서 보였다는 사실도 실제로 그 요청에서 읽고 사용했다는 증거와 구별한다.

읽기·쓰기·실행·외부 행동을 나누어 허용한다

‘이 스킬을 써도 된다’는 말에 여러 행동이 숨어 있다. 메모를 읽을 수 있는가, 새 파일을 쓸 수 있는가, 검사 프로그램을 실행할 수 있는가, 결과를 외부에 보낼 수 있는가다. 안내문 제작에서는 앞의 세 행동이 필요할 수 있지만 마지막 행동은 기본 범위 밖이다. HTML을 만들었다고 웹사이트를 열거나 회원에게 발송하지 않는다.

권한표는 복잡한 보안 표준을 대신하는 문서가 아니라 이번 연습의 경계다. 입력을 읽을 수 있어도 다른 폴더의 모든 파일을 읽을 필요는 없다. 새 결과를 쓸 수 있어도 기존 원본을 덮어쓸 필요는 없다. 검사기를 실행할 수 있어도 패키지를 설치하거나 전역 설정을 고칠 허락이 있는 것은 아니다. 목적에 필요한 행동만 좁게 남긴다.

행동	이 연습의 로컬 경계
읽기	제공한 비민감 메모·선택 스킬·시험 파일
쓰기	새 연습 출력 폴더, 원본 덮어쓰기 금지
실행	읽고 확인한 작은 로컬 검사기
네트워크	검사기에 필요 없음, 외부 연결을 몰래 추가하지 않음
공개·발송·계정·결제	수행하지 않음, 별도 사람 결정과 권한 필요

제품의 샌드박스는 에이전트가 접근할 기술적 경계이며 승인 흐름과 다른 통제라는 점을 [OpenAI 공식 Sandbox 문서, 2026-10-03 확인](#)에서 확인했다. 업무 지시의 ‘하지 않는다’는 문장과 기술적으로 막아 둔 설정은 서로 보완해야 한다. 문장만 있다고 모든 위험이 차단되는 것은 아니고, 파일을 쓸 수 있다고 업무상 허락이 자동 생기는 것도 아니다.

이 책의 제작에서도 로컬 브라우저·Terminal 앱 접근이 보안 정책상 거절된 검사가 있었다. 다른 경로로 몰래 우회해 완료로 바꾸지 않았고, 실제 화면 검토를 미수행으로 남겼다. 샘플 EPUB의 작은 화면·큰 글자 검사는 사용자가 최종 검수로 이관했다. 이관은 검사 없이 PASS를 주라는 뜻이 아니다. 실행 환경이 허용하지 않는 부분을 숨기는 대신 다른 허용된 작업을 계속한 사례다.

비용과 계정은 파일 속 기본값이 아니다

스킬 파일이 로컬에 있다고 모든 실행이 무료인 것은 아니다. 로컬 문서를 읽는 일과 모델을 호출하는 일, 유료 서비스 API를 쓰는 일은 다른 비용 구조를 가질 수 있다. 이 책은 특정 플랜의 현행 요금이나 무료 호출 수를 보장하지 않는다. 자신이 선택한 제품·계정·실행 방식에서 비용과 사용량을 확인해야 한다.

스킬에 ‘필요하면 새 서비스를 결제한다’고 넣어서는 안 된다. 비용이 생기는 행동은 무엇을 왜 쓰고 어느 범위까지 허용할지 먼저 확인한다. API 키나 계정 연결 정보도 독자용 예제에 넣지 않는다. 설정 예시가 필요하다면 실제 비밀값을 복사해 배포하면 안 된다. 이 실습의 작은 검사기는 네트워크와 키를 요구하지 않는다.

기존 시험에서 모델 이름이 기록돼 있다는 것은 그 당시 실제 실행 환경을 알리는 것이다. 독자에게 그 모델을 새로 구매하거나 모든 제품에서 선택 가능하다고 요구하는 것이 아니다. 실행 방식이 바뀌면 조건이 바뀌었음을 남기고 대표 입력을 다시 확인한다. 값이 달라졌는데 옛 시험 보고를 그대로 붙이면 환경을 옮긴 의미가 사라진다.

이 장의 버전 명령에서도 Codex CLI는 경로 별칭을 만들지 못했다는 경고와 함께 버전을 출력했다. 버전을 읽을 수 있었다고 모든 준비가 성공했다고 보고하지 않는다. 설정을 바꾸어 경고를 없애지 않았으며, 명령 출력과 경고를 함께 기록했다. 환경 확인의 결과는 ‘있음/없음’ 두 값보다 구체적일 수 있다.

같은 파일 이름보다 기준 폴더가 중요하다

`안내문.html`이 있는데 파일 없음이 나온다면 먼저 현재 폴더를 본다. 같은 이름의 파일이 다른 위치에 있을 수 있다. HTML 안의 `images/logo.png`는 HTML 파일이 있는 폴더를 기준으로 찾지만, 실행 명령의 검사 프로그램 경로는 명령을 실행하는 현재 폴더를 기준으로 해석된다. 두 기준을 혼동하면 보조 파일을 엉뚱한 위치로 옮기기 쉽다.

개인 컴퓨터의 전체 경로를 예제에 넣으면 다른 사람이 그대로 실행할 수 없다. 이 장의 연습 묶음은 상대 경로와 명시적인 기준 폴더를 사용한다. 폴더 이름에 공백이 있을 때에는 경로를 따옴표로 감싼다. 읽기 쉽게 줄을 나누는 표기와 실제 셸이 읽는 문법은 다르므로, 명령을 복사할 때 사용하는 셸을 먼저 확인한다.

```
python3 check_handout.py "연습 사본/안내문.html"
```

위 명령은 검사기 파일이 있는 연습 폴더에서 시작한다는 예시다. 이번 제작에서는 공백이 든 폴더와 한글 파일 이름으로 사본을 만들고, macOS zsh에서 따옴표로 감싼 경로의 기본 검사 PASS를 실제 확인했다. 이 결과는 경로 표기와 로컬 검사 실행의 증거이며, Windows나 다른 셸에서도 같은 명령이 된다는 시험은 아니다.

경로가 맞아도 읽기 권한이나 파일 형식이 다를 수 있다. 허락 없이 권한을 넓히지 말고 어떤 파일에서 어떤 오류가 났는지 기록한다. 파일에 접근할 수 없으면 읽은 것처럼 내용을 보고하지 않는다. 필요하다면 사용자가 비민감 사본을 제공하거나 허용된 위치를 선택해야 한다. 이 책은 접근 거절을 풀기 위한 전역 보안 설정 변경을 기본 해결책으로 제시하지 않는다.

다른 환경에서는 무엇을 다시 확인할까

스킬 문서의 모양이 같아도 제품이 추가 파일을 읽는 방법, 명령을 실행하는 도구, 승인 화면, 셀은 다를 수 있다. 처음에는 정상 입력 하나와 작은 실패 입력 하나로 확인한다. E01은 근거 있는 제목과 빈 주제, E02는 제한된 목표와 자료 대기, E03은 단순 메모와 누락 로컬 이미지가 출발점이 될 수 있다. 필요한 입력 전체와 비호출 검사는 그 뒤 범위에 맞게 넓힌다.

여기서 비호출이란 해당 스킬의 일이 아닌 요청에서 스킬을 사용하지 않는지 보는 것이다. E03의 설명이 넓어지면 일정 요약이나 서버 개발 요청까지 안내문으로 만들 수 있다. 파일을 옮긴 뒤에는 결과의 품질뿐 아니라 어떤 요청에서 시작하는지도 확인해야 한다. 다른 환경을 시험했다면 제품·버전·읽은 파일·입력·실제 출력·기대 조건을 함께 남긴다.

이 장에서 만든 로컬 검증 묶음은 파일 누락과 내용 변경을 찾는 데 한정된다. 선택된 세 스킬의 사본과 검사기, 기준 해시 목록, 확인 스크립트를 함께 둔다. 해시는 파일의 내용이 기준과 같은지 비교하는 값이다. **같은 해시가 안전성·독창성·권리·AI 행동을 보장하는 것은 아니다.** 검증 묶음을 받는 사람은 코드를 읽고 허용할 행동을 정한 뒤 실행한다.

이번 실제 고정 시험에서는 정상 묶음, 필수 파일 하나가 빠진 사본, 스킬 문장 하나가 바뀐 사본을 비교했다. 정상은 일치, 두 실패 사본은 각각 누락과 해시 불일치를 보고했다. 새 AI 세션의 제품 호환 시험은 아니다. 로컬 파일을 옮기는 중 구성과 내용이 바뀌었는지 확인하는 증거다.

실패했을 때 전부 지우거나 전부 다시 돌리지 않는다

복사 도중 파일이 빠졌다면 선택한 원본에서 그 파일만 복구한다. 출처를 모르는 다른 스킬의 비슷한 파일로 채우지 않는다. 기준 목록과 맞는지 확인한 다음 영향을 받는 검사를 다시 한다. 원고나 스킬을 의도적으로 고쳤다면 기준 해시도 바뀌어야 하지만, 해시부터 바꾸어 실패를 없애면 검증의 의미가 없다. 수정 이유와 새 시험 범위를 먼저 정한다.

이번 **교육용 가상 사례**에서 담당자 도현은 동료에게 E03 안내문 연습을 넘긴다. 동료의 환경에서는 파이썬 명령을 찾지 못한다. 도현은 회사 정책을 확인하지 않은 채 설치 명령을 보내지 않는다. 동료가 사용할 수 있는 실행 환경과 설치 허락을 먼저 확인하고, 당장은 HTML 초안을 읽는 작업과 프로그램 검사를 구분해 둔다.

가상 상황에서 허용된 실행기가 확인됐다고 설정하자. 다음 실패는 없는 로고 파일이다. 도현은 실행기를 다시 설치하지 않는다. 같은 HTML과 같은 검사기로 로컬 파일의 위치를 확인하고, 로고가 필수인지 사람이 결정하도록 한다. 제품 설치 문제와 자료 누락 문제는 다른 원인이므로 복구도 다르다.

동료에게 공유할 파일도 한정한다. 실제 회사 자료와 개인 실행 로그를 통째로 묶지 않고 비민감한 연습 사본과 검사 기록만 준비한다. 작성자의 개인 경로·키·고객 정보가 있는지 확인한다. 아직 외부에 전달할 승인이 없다면 로컬 폴더에서 검토를 끝낸다. ‘공유 준비 완료’는 실제 전송을 했다는 뜻이 아니다.

환경표와 중단 안내를 함께 남긴다

다음은 이번 제작의 **저자 작성 실행 준비 카드**다. 실제 확인과 미수행을 같이 적었다. 권한 경계는 이번 교육용 로컬 작업의 범위이며 모든 조직에 적용되는 보안 규정이 아니다.

항목	작성 기록
실제 환경	macOS·Codex0.145.0, Python 실행기 별도 기록
선택 파일	E01 v5·E02 v3·E03 v2와 E03 작은 검사기
허용 행동	비민감 자료 읽기, 새 사본 쓰기, 로컬 검사
제외 행동	전역 설치·계정·결제·외부 공개·Git 변경
고정 검사	정상/누락/내용변경 묶음, 공백 경로의 HTML
미수행	타 OS·타 제품의 AI 호출, 실제 리더·기기
중단 때	오류·선택 버전 보존, 필요한 권한을 별도 확인

복구 안내에는 ‘다시 실행’만 적지 않는다. 어떤 원본에서 어떤 파일을 복구하고 무엇을 다시 확인할지 적는다. 스킬의 의미를 고쳤다면 정상·실패 입력의 행동 시험이 필요하다. 파일만 누락됐다면 먼저 파일 구성 검사를 한다. 작업을 반복할수록 이 구분이 불필요한 전체 재실행과 과도한 권한 확대를 줄인다. 실제 절약 시간을 측정하지 않았으므로 수치 성과로 표현하지 않는다.

내가 적을 항목	내용
제품·버전·운영체제·셀	
사용할 실행기와 의존 도구	
읽을 자료·쓸 위치·기준 폴더	
허용한 실행과 제외한 외부 행동	
비용·계정·설치가 필요할 때 결정자	
실제 확인한 입력·결과·한계	
오류를 보존하고 복구할 순서	
옳긴 뒤 다시 확인할 정상·실패 요청	

이 장의 작업물은 설치가 끝났다는 인증서가 아니다. 환경표·권한표·복구 안내와 실제 파일 구성 검증을 묶어, 어디까지 확인했는지 다른 사람이 알게 만든 기록이다. 12장에서는 이 같은 파일 중심의 약속을 여러 제작 역할 사이에 연결한다. 스킬이 늘어나도 외부 행동의 허락이나 사람의 최종 판단을 자동으로 넘겨주지 않는다는 기준은 그대로 유지한다.

PART 5 / 다음 제작으로

여러 스킬로 결과물 완성하기

이제 여러 역할의 출력을 이어 작은 가이드를 완주한다. 일곱 역할을 외우기보다 받은 파일과 다음 역할의 필요가 맞는지 확인한다. 여덟 쪽 가이드를 실제 변환하고 실패 사본과 새 실행 재개를 시험한다.

남길 것은 가이드와 재현 패키지, 정확한 미해결 목록이다. 기술 결과가 열렸다고 공개·판매가 승인되는 것은 아니다. 다음 사람이 같은 입력을 찾아 다시 만드는 데 필요한 정보를 끝에 남긴다.

12 / 여러 스킬로 결과물 완성하기

여러 스킬을 하나의 일로 연결하기

언제 스킬을 나누고 어떻게 결과를 넘기는가?

이 장에서 남길 것: 7개 역할의 파일 인계표

원고를 썼고 표지도 만들었는데 PDF의 책 제목이 다르다. 편집자는 수정한 원고를 보냈지만 조판은 예전 파일을 읽었다. 검사 보고에는 PASS라고 적혀 있으나 어느 PDF를 검사했는지 알 수 없다. 각 작업이 잘 끝났다고 해서 전체 결과가 이어지는 것은 아니다. 앞사람이 넘긴 것과 뒷사람이 받은 것이 같아야 한다.

이번 장은 스킬을 더 많이 실행하는 법보다 **스킬 사이에서 어떤 파일과 판단을 넘길지** 다룬다. 전자책 제작을 예로 들지만 대형 출판 자동화 시스템을 새로 만들지는 않는다. 독자가 작은 업무 가이드를 완주할 수 있도록 일곱 역할의 경계와 파일 인계표를 만든다. 실제 8~12쪽 제작은 13장에서 수행한다. 역할을 나눈다고 병렬 에이전트·새 채팅·외부 공개를 자동으로 시작하지 않는다.

먼저, 끝났다는 말 대신 받은 파일을 보자

역할의 연결은 결과 파일의 연결이다

‘집필 완료’라는 보고보다 선택 원고를, ‘검수 완료’라는 보고보다 검사한 출력의 이름과 해시를 확인한다.

이 책에서 참고하는 일곱 제작 스킬은 Coreline-AI 기존 프로젝트의 로컬 원본이다. 기획과 진행을 잇는 입구, 원고 집필, 편집, 근거 검토, 시각 자산, 형식 제작, 결과 검증으로 역할을 나눴다. 이는 저자가 만든 제작 구조이지 OpenAI 제품의 공식 일곱 단계나 Agent Skills 표준의 필수 구성은 아니다. 원본 파일과 그때의 해시를 읽고 역할을 참고했으며, 이 장을 위해 원본 스킬 전체를 새로 실행한 것은 아니다.

스킬 하나가 지시·참고 자료·선택적 도구를 묶는다는 점과 업무 하나에 집중하는 작성 원칙은 [OpenAI 공식 Build skills, 2026-10-03 확인](#)에서 확인했다. 역할을 나누는 구체적 기준은 이 책의 제작 경험에서 설명한다. 원고를 쓰는 책임과 PDF를 만드는 책임이 다르면, 수정할 파일과 다시 확인할 대상도 분명해진다.

진행 입구 · 현재 파일과 다음 작업 연결

원본 보존 · 미해결 · 사람 결정 유지

1 집필

독자용 원고

2 편집

선택 편집본

3 근거

주장 · 권리 기록

4 디자인

도해 · 메타데이터

5 빌드

변환 파일 · 입력 기록

6 검증

정확한 출력 · 미수행

기술 PASS를 공개 · 판매 승인으로 넘기지 않는다

그림 12-1. 여섯 전문 역할과 전체 진행을 잇는 입구, 총 일곱 역할이다. 저자의 설계이며 자동 병렬 실행이나 제품 기능 화면이 아니다.

일을 나누는 이유는 담당자의 수가 아니다

혼자 일해도 역할은 나눌 수 있다. 오전에는 집필자, 오후에는 편집자, 마지막에는 검수자처럼 같은 사람이 다른 기준으로 결과를 읽는다. 중요한 것은 일곱 사람이 있거나 일곱 AI가 동시에 실행되는지가 아니다. 각 역할이 받는 재료, 바꿀 수 있는 파일, 넘길 결과, 넘기지 못할 승인을 구분하는 것이다.

집필자는 설명과 사례를 만든다. 편집자는 완성된 초고의 중복과 순서, 용어를 고친다. 빌드는 선택된 원고를 읽어 정해진 형식으로 바꾼다. 빌드 중 설명이 이상하다고 본문을 몰래 고치면, 선택 원고와 PDF가 달라져 누

가 무엇을 검토했는지 알기 어렵다. 그런 문제는 편집 역할로 돌려 새 원고를 선택한 뒤 다시 빌드해야 한다.

검증 역할도 결과를 직접 고치지 않는다. 출력에서 오류를 찾고 어떤 파일에서 무엇을 발견했는지 보고한다. 수정은 해당 원고·자산·변환 코드의 역할에서 한다. 불편해 보이지만 검사 대상이 검사 중 바뀌지 않아야 보고의 의미가 남는다. 혼자 하더라도 역할을 오갈 때 사본과 기록을 남기는 이유다.

진행 관리 입구는 모든 상세 업무를 대신하지 않는다. 실제 현재 파일과 진행 상태를 대조하고 다음 역할을 정한다. 이전 문서에 미착수라고 적혀 있어도 실제 원고가 있으면 먼저 그 파일을 읽는다. 반대로 파일 이름만 있다고 완성으로 표시하지 않는다. 원고·자체 검토·기술 검사·사람 수용을 각각 확인한다.

일곱 역할에는 서로 다른 끝이 있다

아래는 기존 일곱 스킬의 이름과 이 실습에서 읽을 역할이다. 이름은 참조한 실제 파일의 **name**과 대응한다. 독자가 모두 설치해야 한다는 목록은 아니다. 원본 코드를 통째로 받기보다 각 역할이 어떤 결과를 넘기는지 먼저 이해한다.

실제 스킬 이름	이 책에서 참고한 역할
orchestrate-ebook-product ion	현재 상태·보존·다음 작업·사람 결정 연결
write-ebook-manuscript	독자용 초고와 장별 검토 기록
edit-ebook-manuscript	원본을 보존한 편집본과 장간 정합성
audit-ebook-evidence	사실 근거와 권리·출처의 별도 검토
design-ebook-assets	표지·도해·글꼴·메타데이터·제작 이력
build-ebook-formats	선택 입력에서 Markdown·PDF·EPUB 변환
validate-ebook-release	정확한 출력 검사·재현 비교·허용 목록 패키지

끝의 뜻도 다르다. 집필의 끝은 독자용 본문이 저장되고 자체 검토가 끝났다는 뜻이다. 편집의 끝은 문장과 장간 연결을 검토한 새 원고가 선택됐다는 뜻이다. 근거 검토는 출처와 미해결을 연결하되 법률상 완전한 권리 보장을 대신하지 않는다. 형식 제작의 끝은 파일이 생겼다는 뜻이고, 검증의 끝은 지정 범위의 검사 결과가 그 파일에 연결됐다는 뜻이다.

가장 마지막 기술 PASS도 저자의 출판 승인과 다르다. 공개 범위와 판매 여부, 최종 원고·표지의 선택은 사람이 결정한다. 역할을 연결할수록 이 경계를 흐리기 쉽다. ‘마지막 단계니까 이제 업로드’라는 동작을 기본값으로 넣지 않는다. 이번 책과 종합 실습은 로컬 제작까지만 수행한다.

인계표에는 이름보다 경로와 상태가 필요하다

‘원고 전달’이라는 항목만 쓰면 파일을 받은 사람은 여러 버전 중 하나를 골라야 한다. **draft01**, **draft02**, **edited01**이 함께 있다면 어떤 것이 선택 원고인지 명시한다. 이름에 **final**이 들어 있어도 실제로 최종 승인된 파일이라는 뜻은 아니다. 전달하는 쪽이 명시적인 선택 목록을 만들고, 받는 쪽이 그 목록과 파일을 비교한다.

이 책은 이를 ‘파일 인계 계약’이라고 부른다. 거창한 외부 표준이 아니라 다음 작업의 입력과 출력에 대한 작은 약속이다. 최소한 입력 파일, 결과 파일, 검토 상태, 미해결, 다시 시작할 곳을 남긴다. 중요한 자료에는 내용 해시를 붙여 같은 이름의 파일이 바뀌었는지 비교한다. 해시는 의미나 권리의 승인값이 아니라 내용 동일성 확인값이다.

계약에는 하지 않을 행동도 넣는다. 빌드 역할은 선택 원고를 쓰지 않는다. 검사 역할은 PDF를 고치지 않는다. 다음 역할이 파일을 읽을 수 있어도 이전 역할의 미해결을 없앨 수는 없다. 모르는 출처가 있으면 확인하거나 그 주장을 빼는 편집 결정을 다시 해야 한다. 근거 확인을 끝내지 않고 표지만 예쁘게 만드는 것은 진행이 아니라 누락을 가리는 일일 수 있다.

인계	함께 넘길 핵심 정보
집필 → 편집	선택 초고, 장 범위, 실제/가상 구분, 미해결
편집 → 근거·자산 검토	새 편집본, 변경 이유, 주장·그림 위치
근거·자산 → 형식 제작	선택 자산과 출처, 메타데이터, 미해결 경계
형식 제작 → 검증	입력 목록·해시, 출력·해시, 실제 빌드 기록
검증 → 사람 검토	정확한 파일별 판정, 실제 미수행, 공개 승인 대기

인계표는 일을 순서대로 나열하는 계획표와 다르다. 어떤 파일이 현재 다음 단계의 입력인지 지정한다. 단계가 끝날 때 파일이 바뀌면 이전 입력을 남기고 새 선택을 만든다. 나중에 PDF의 제목이 왜 달라졌는지 보려면 당시 메타데이터와 당시 빌드 입력이 있어야 한다.

형식이 다르면 멈추고 번역한다

앞 역할은 Markdown 원고를 넘겼는데 다음 역할은 JSON만 읽는다면 그대로 이어지지 않는다. 파일 확장자를 바꾸는 것만으로 내용의 구조가 달라지지도 않는다. 변환 단계가 필요한지, 다음 역할이 Markdown을 읽도록 설계할지 결정한다. 받은 쪽이 빈 입력을 정상 입력으로 취급하면 결과 파일은 생겨도 내용은 틀릴 수 있다.

확장자뿐 아니라 항목의 뜻도 맞아야 한다. 한 역할의 ‘완료’가 초고 저장이고 다른 역할의 ‘완료’가 저자 승인이라면 같은 값으로 연결할 수 없다. **DRAFT**와 **APPROVED** 같은 영어 표기를 늘리는 것이 해결은 아니다. 사

람이 읽을 말로 ‘초고 저장·자체 검토’와 ‘최종 승인’을 나누고 파일 상태에 붙인다.

가상 예시에서는 작성자가 그림 파일 이름을 목록에 넣었지만 디자이너가 아직 파일을 만들지 않았다. 빌드는 그림의 이름을 실제 파일처럼 처리하면 안 된다. 자료표는 계획이고 실재 파일은 다른 조건이다. 누락을 발견하면 해당 단계가 보류되고, 그 그림이 설명에 필수인지 편집자가 판단한다. 삭제하기로 했다면 원고의 참조·캡션·대체 텍스트도 함께 고친다.

이번 장의 로컬 인계 검사 연습에서는 입력 형식을 Markdown으로 지정하고 실제 파일과 해시를 대조했다. 정상 인계는 연결됐고, 없는 파일을 적은 사본은 누락, JSON을 기대한다고 바꾼 사본은 형식 불일치를 보고했다. 프로그램으로 확인한 고정 시험이다. 일곱 스킬이 에이전트로 자동 실행돼 가이드를 제작했다는 결과는 아니다.

한 부분을 고치면 어디까지 돌아갈까

원고의 제목을 고치면 표지 글자, 제목 페이지, PDF 메타데이터, EPUB 제목, 파일 목록을 다시 확인해야 한다. 본문 그림 한 점을 바꾸면 그림 출처·원고 설명·PDF 렌더·EPUB 자산이 영향을 받는다. 검사 보고가 그대로 있어도 바뀐 출력의 해시와 다르면 그 보고를 현재 판정으로 사용할 수 없다.

반대로 모든 변경이 모든 작업의 반복을 뜻하지는 않는다. 진행 기록의 이탈자를 고쳤는데 독자용 입력이 그대로라면 PDF를 다시 만들 필요가 없을 수 있다. 그러나 무엇이 빌드 입력인지 명확해야 이 판단을 할 수 있다. 파일을 많이 나눈 프로젝트일수록 명시적인 입력 목록이 중요해진다. 입력으로 동결한 파일이 바뀌면 다음 결과를 새로 만들고 검토한다.

스킬의 지시를 고친 경우에는 문서 변환 재현성만 보면 안 된다. 생성형 출력은 새 요청의 조건과 세션에 따라 달라질 수 있다. 같은 원고로 같은 PDF가 만들어지는 확인과, 같은 스킬이 새 입력에서도 업무 경계를 지키는 확인은 다른 시험이다. 두 결과를 한 ‘재현성 PASS’로 묶지 않는다.

작업이 멈췄을 때도 마찬가지다. 출력 폴더에 PDF가 남아 있더라도 빌드 기록이 완료됐는지, 파일이 열리는지, 정확한 입력을 사용했는지 확인한다. 반쯤 만들어진 파일을 최신이라고 선택하지 않는다. 이전에 검증한 결과는 그대로 두고 새 실행 폴더에서 이어 간다. 13장에서 실제 중단과 재개를 작게 시험한다.

가상 편집자는 ‘최신 파일’을 고르지 않았다

이번 **교육용 가상 사례**의 편집자 민아는 자료 정리 업무 가이드를 준비한다. 폴더에는 초고 두 개와 검토본 하나가 있다. 가장 최근 수정된 파일은 검토자의 개인 메모이며 독자용 원고가 아니다. 민아가 ‘최근 파일을 쓰자’고 선택하면 내부 의견이 PDF 본문에 들어갈 수 있다.

민아는 원고를 실제 열어 읽고 독자용 편집본을 선택 목록에 적는다. 근거 검토자는 그 파일의 정책 표현을 가상 규칙으로 표시했는지 확인한다. 디자인 담당자는 설명 도해와 대체 텍스트를 연결한다. 빌드 담당자는 선택 목록에 없는 개인 메모를 읽지 않는다. 검증자는 출력 PDF와 사용한 선택 목록의 해시를 함께 보고한다.

가상 상황에서 도해가 한 점 빠졌다고 설정하자. 민아는 빈 상자를 대신 넣어 완성이라고 하지 않는다. 그 관계를 문장만으로 설명할 수 있는지 판단하고, 필요하면 그림을 완성할 때까지 빌드를 보류한다. 다시 빌드할 때 이전 PDF는 남기고 새 출력으로 저장한다. 사람이 정한 삭제나 대체 결정 없이 도구가 임의로 설명을 바꾸지 않는다.

그 뒤 검증 보고가 나와도 민아는 출판을 자동으로 시작하지 않는다. 실제 작은 화면과 큰 글자 검사가 남았고, 저자가 최종 원고를 아직 선택하지 않았을 수 있다. 인계표의 마지막 칸은 공개 완료가 아니라 사람이 결정할 목록이다. 기술 역할이 할 수 있는 일과 최종 책임자가 할 일을 구분한 결과다.

내 업무의 인계 한 줄을 먼저 적는다

아래는 이번 장을 위해 저자가 작성한 **소형 가이드 인계 예시**다. 파일 이름은 다음 장의 실습 계약에 대응할 교육용 이름이다. 아직 이 표 자체로 가이드 제작이 완료됐다고 말하지 않는다. 현재 실제 확인한 것은 별도의 인계 검사 연습 파일이다.

역할	받을 것과 남길 것
진행 입구	목적·제외·현재 상태 → 다음 작업과 미해결
집필	가상 업무 메모 → 독자용 Markdown과 확인 질문
편집	선택 초고 → 새 편집본과 변경 이유
근거 검토	편집본 → 실제/가상 구분·출처·사람 판단
디자인	설명할 관계 → 도해·alt·제작 기록
형식 제작	명시 선택 목록 → 새 PDF와 입력/출력 기록
검증	정확한 PDF → 구조·시각·재빌드·미수행 기록

일곱 역할을 자기 업무에 그대로 복사할 필요는 없다. 한 페이지 안내문이 라면 집필·내용 검토·파일 검사·사람 승인을 구분하는 것만으로도 충분 할 수 있다. 업무가 커져 결과를 주고받는 이유가 생겼을 때 나눈다. 스킬 숫자가 늘어났다는 사실을 전문성이나 성과 지표로 삼지 않는다.

지금 맡는 일에서 두 역할 사이 한 줄을 골라 다음 양식을 채워 보자. 입력 과 결과의 파일 이름을 적을 수 없으면 아직 인계가 모호한 것이다. 사람 이 해야 할 결정을 결과 파일의 존재로 대신하지 않는지도 함께 확인한다.

내가 적을 항목	내용
앞 역할과 다음 역할의 이름	
실제 넘길 입력 파일과 형식	
선택 버전·내용 동일성 확인 방 법	
다음 역할이 바꿀 수 있는 것	
다음 역할이 하지 않을 행동	
결과 파일·검토 상태·미해결	
입력이 없거나 다를 때 멈출 곳	
수정 뒤 다시 볼 하위 결과	

이번 장의 결과는 역할이 읽을 파일과 넘길 상태를 정한 인계표다. 13장 에서는 작은 가이드를 실제 만들고, 그림 누락·입력 변경·의도적 중단과 새 실행 재개를 확인한다. 단계 수보다 원본과 상태를 보존하며 다시 시작 할 수 있는지가 중요하다.

13 / 여러 스킬로 결과물 완성하기

작은 업무 가이드를 끝까지 제작하기

원고부터 PDF까지 실제로 어떻게 완주하는가?

이 장에서 남길 것: 8~12쪽 소형 업무 가이드

이번에는 열린 파일로 끝을 확인한다.

원고를 썼다는 말과 읽을 수 있는 가이드를 만들었다는 말 사이에는 편집, 그림 연결, 변환, 검사가 있다. 그 거리를 작은 결과물 하나로 건너가 보자.

먼저, 실제로 만든 여덟 쪽을 펼친다

이번 종합 실습의 결과는 「팀 검토 파일 준비하기 — 교육용 미니 가이드」다. 저자는 Coreline-AI이며, A5 PDF 실측 여덟 쪽이다. 원고 여덟 파일을 직접 작성하고 설명 도해 한 점과 작성 카드를 넣었다. 같은 선택 입력으로 PDF·EPUB·통합 Markdown을 로컬에서 생성했다. 파일이 있다고만 말하지 않고 다시 열어 메타데이터와 쪽수, 글꼴, 내용 추출을 검사하고 PDF 여덟 쪽을 모두 렌더해 읽었다.

여기서 무엇이 실제인지 먼저 구별하자. 가이드의 인물 지윤, 팀의 이름 규칙, 검토 상황은 **교육용 가상 사례**다. 실제 회사의 문서 절차를 취재한 사례가 아니다. 반면 원고 작성, 입력 파일의 해시 확인, Python 변환,

PDF 검사, 누락과 재개 시험은 이 책을 만드는 과정에서 실제 수행했다. 일곱 스킬을 각각 새로운 AI 세션에서 자동 호출해 완성한 시험은 하지 않았다. 12장의 역할 구분에 따라 저자가 단계별 책임을 맡고 로컬 도구로 변환한 실습이다.

이 구별이 중요한 이유는 독자가 따라 할 대상을 정확히 알게 하기 위해서다. 실제 변환 과정을 따라 하면 자신의 폴더에서도 같은 입력이 같은 출력을 만드는지 확인할 수 있다. 그러나 지윤의 이름 규칙을 실제 팀 규정으로 쓰거나, 이 시험을 AI의 업무 판단 능력 검증으로 받아들이면 다른 문제다. 이 장이 확인한 것은 작은 문서 제작의 연결과 중단 경계이지 모든 조직·모델·운영체제에서의 성공이 아니다.

처음부터 두꺼운 책을 만들지 않은 것도 같은 이유다. 결과가 너무 커지면 잘못된 입력 하나, 읽히지 않는 그림 하나, 거의 빈 끝 페이지 하나를 놓치기 쉽다. 여덟 쪽이면 원고와 PDF를 나란히 펼쳐 빠진 문장과 새로 생긴 문제를 찾을 수 있다. 작게 만드는 것은 연습을 대충 한다는 뜻이 아니라 검토 가능한 크기로 책임을 제한한다는 뜻이다.

범위를 작게 정하면 끝을 설명할 수 있다

실습의 목적은 팀 검토 파일을 준비하는 방법을 설명하는 것이다. 내용은 받은 메모 보존, 새 사본 준비, 이름 예시, 메모 충돌, 검토 질문, 누락 처리, 재개 기록, 자기 업무 적용으로 나뉘었다. 여덟 주제는 각각 독자가 해야 할 행동 하나를 설명한다. 내용을 여덟 칸으로 나누었다는 이유만으로 여덟 개의 스킬이 필요하지는 않다. 원고 파일의 단위와 스킬의 역할은 다르다.

포함하지 않을 것도 먼저 적었다. 실제 공유 폴더를 찾아 들어가기, 회사의 개인정보 읽기, 문서 발송, 외부 출판, 전역 스킬 설치, 저장소 변경은 하지 않는다. 날짜와 역할 이름은 가상값이라고 표시한다. 원고를 예쁘게

만들었다고 공유 허락까지 생기지 않는다는 문장을 가이드 초반에 놓았다. 뒤쪽에서만 주의사항을 말하면 독자가 앞부분의 작업을 실제 실행 승인으로 오해할 수 있기 때문이다.

이 실습의 완료는 ‘검토 가능한 새 문서와 기록이 남음’이다. PDF 변환이 성공했다고 사실 검토와 출판 승인을 한꺼번에 완료로 표시하지 않는다. 문서 변환, 구조 검사, 자체 시각 검토, 실제 리더 검사, 저자 수용을 별도 상태로 적는다. 이 작은 가이드에서도 EPUB 구조와 EPUBCheck는 확인했지만 실제 작은 화면과 큰 글자 리더 검사는 남겨 두었다.

스킬은 지시와 자료, 선택적인 스크립트를 묶는 형식이다. 폴더와 파일을 연결했다고 필요한 프로그램이나 권한이 모두 제공되는 것은 아니다. 공식 설명에서도 SKILL.md와 선택적인 자료·스크립트를 구별한다. 이번 실습의 Python 조판기는 별도로 준비한 로컬 제작 도구이며 스킬 자체의 내장 기능으로 설명하지 않는다. [OpenAI Build skills](#), 2026-10-03 확인.

예산은 여덟에서 열두 쪽으로 정했지만, 빈 페이지를 넣어 그 숫자에 맞추지 않았다. 원하는 분량은 독자가 한 번에 검토할 범위를 정하는 조건이다. 열세 쪽이 나왔다면 먼저 내용과 조판을 보고 이유를 판단해야 한다. 글자를 갑자기 작게 만들거나 여백을 없애는 것은 독자의 읽기 조건을 몰래 바꾸는 방법이다. 숫자를 맞추는 것보다 읽을 내용과 이유가 남는지가 먼저다.

원고와 그림을 한 번에 넘기지 않는다

가이드 원고는 여덟 Markdown 파일로 저장했다. 첫 파일은 목적과 제외 범위를, 둘째 파일은 원본과 작업본의 차이를 설명한다. 셋째에서는 역할·날짜·내용을 넣은 가상 이름 예시를 보여 준다. 넷째의 자료 충돌은 어느 규칙이 더 좋아 보이는지 AI가 고르는 장면이 아니다. 두 메모의 차이를 사람에게 질문하고, 답을 받은 뒤 새 검토본을 만드는 과정이다.

그림은 원본 보존→새 검토본→사람 결정의 세 단계를 보여 준다. 보존 폴더와 작업 폴더만 그리면 마지막 단계가 자동 발송처럼 보일 수 있어 사람 결정을 함께 넣었다. 그림의 질문은 ‘새 파일이 생겼는데 왜 아직 결정이 남는가?’다. 이 관계를 이해하도록 쓰는 그림이며 제품 화면을 흉내 낸 인터페이스가 아니다. 본문이 설명하는 범위와 그림의 마지막 행동이 일치해야 한다.

원고 파일, 그림, 글꼴, 스타일, 변환 스크립트, 입력 메모, 실행 안내를 선택 목록에 명시했다. 각 파일의 SHA256 값은 같은 내용을 사용했는지 확인할 긴 식별값이다. 독자가 외울 필요는 없다. 목록에 있는 값과 실제 파일의 값을 도구가 대조한다는 원리만 이해하면 된다. 본문을 읽어야 할 사람과 동일성을 계산하는 프로그램이 서로 다른 검토를 맡는 것이다.

선택 목록에는 버전 이름뿐 아니라 실제 파일 경로가 들어 있다. 폴더 안에서 가장 최근 파일을 고르는 방식은 쓰지 않았다. 이름이 비슷한 옛 원고나 내부 메모가 섞여 있어도 지정한 여덟 원고만 사용한다. 그림 이름만 목록에 적고 실제 파일이 없으면 검증을 통과하지 못한다. 원고를 몰래 줄이거나 임시 그림을 넣어 결과를 만들지 않는다.

가상 업무 원고 · 실제 작성 파일

입력과 그림 선택 → 해시 대조

실제 로컬 변환 · 새 실행 폴더

PDF 8쪽 / EPUB / Markdown

출력 재독 · 실패 사본 · 새 실행 재개

구조와 시각 확인 / 같은 입력 결과 비교

기기 검수 · 저자 수용은 별도다

그림 13-1. 가상 업무의 원고를 실제 파일로 작성하고, 선택 입력 검사→새 출력→정확한 출력 검토로 연결했다. 실패 사본은 별도 폴더에 남겼다. 제품 실행 화면이나 일급 스킬 자동 호출 기록이 아니다.

그림을 바꾸었다면 그림만 열어 보는 것으로 끝내지 않는다. 본문의 뜻, 캡션과 대체 텍스트, PDF에 들어간 크기, EPUB의 자산 연결을 다시 본다. 원고에서는 잘 보이는 큰 글자가 A5에 넣었을 때 작아질 수 있다. 파일이 있다는 검사와 실제 독자가 읽을 수 있는 크기의 검토를 나누는 이유다.

새 실행 폴더에서 변환하고 결과를 다시 읽는다

독자용 실습 묶음의 E04 폴더에는 원고, 도해, 글꼴 안내, 선택 목록, 두 Python 파일이 들어 있다. 다음 명령은 그 폴더 안에서 실행하는 예시다. 확인된 Python에 ReportLab과 pypdf가 있어야 한다. 필요한 프로그램을 자동 설치하거나 추가 비용을 승인하는 명령이 아니다. 자신의 환경에 도구가 없다면 먼저 무엇을 설치할지 확인한다.

```
python3 build/build_editorial_book_v3.py \  
--manifest manifest.json \  
--run-dir builds/new-run
```

역슬래시는 다음 줄까지 하나의 명령으로 이어 쓰기 위한 표시다. new-run은 아직 없는 실행 폴더 이름으로 정한다. 같은 폴더가 이미 있으면 다른 이름을 사용한다. 기존 결과를 지워 다시 만드는 대신 이전 결과와 새로운 결과를 비교할 수 있게 남긴다. 실제 제작에서는 확인된 Python 3.12.14 환경을 사용했다. 이 명령을 모든 환경에서 시험한 것은 아니다.

변환기는 먼저 입력 해시를 검사한다. 통과하면 새 실행 폴더에 book.pdf, book.epub, combined.md와 빌드 기록을 저장한다. 기록에는 사용한 선택 목록, 출력 파일 해시, 쪽수와 각 내용의 범위가 들어 있다. 출력 파일의 존재만으로 성공을 판단하지 않도록 입력과 결과를 함께 남긴다. 같은 이름의 PDF라도 내용이 다르면 이전 검사 기록을 가져다 붙일 수 없다.

처음 변환한 판은 열두 쪽이었다. 숫자상으로는 목표 범위였지만 실제로 펼쳐 보니 네 개의 끝 페이지에 짧은 결론만 남았다. 이 결과를 ‘분량 달성’으로 선택하지 않았다. 가이드 앞부분에서 이미 정한 공개·승인 경계를 뒤에서 되풀이하는 문장을 정리하고, 파일 이름의 승인 여부 설명은 표 앞 문단에 합쳤다. 중요한 규칙을 빼서 짧게 만든 것이 아니라 반복 위치를 고친 것이다.

새 편집판은 여덟 쪽이 됐다. 본문 11pt와 실제 보통·굵은 글꼴을 유지했다. 원고와 조판을 다시 읽고 여덟 쪽의 접촉 시트로 전체 균형을 확인했다. 첫 판과 실패 기록은 보존했다. 이 변화는 가독성을 위한 편집이고 AI

가 업무 시간을 줄였다는 성과 수치가 아니다. 같은 가상 업무를 다른 배치로 설명한 두 판의 실제 쪽수다.

출력 검토는 내용과 형태를 번갈아 보는 작업이다. 날짜가 가상값이라고 표시됐는지, 미해결 승인자를 확인한 문장이 없는지 읽은 다음 표가 나뉘거나 그림 글자가 잘리지 않았는지 본다. PDF 메타데이터의 제목과 저자도 대조한다. 표지가 없어도 파일 속성에는 다른 책 이름이 들어갈 수 있기 때문이다. EPUBCheck의 오류·경고가 없다는 결과는 실제 리더의 사용성을 대신하지 않는다고 기록했다.

세 가지 실패를 일부러 작게 만든다

첫 시험은 그림 누락이다. 정상 입력을 격리 폴더에 복사한 뒤 flow.svg만 제거했다. 변환기는 파일 검사에서 종료 값 1을 반환했고 PDF를 만들지 않았다. 이 시험은 실제 로컬 실행이다. 원고에 그림이 빠진 상태를 발견했다는 뜻이지, 그 그림이 업무상 필요 없다는 뜻은 아니다. 원본을 복구하거나 사람이 삭제를 결정해 원고와 선택 목록을 새로 만드는 두 경로가 남는다.

둘째는 낡은 선택 목록이다. 격리 사본의 입력 메모에 문장 하나를 추가하고 목록의 해시는 그대로 두었다. 변환은 종료 값 1로 중단됐고 PDF가 생기지 않았다. 수정한 메모를 무시하고 옛 결과를 만들지 않도록 하는 검사다. 새 내용이 필요한 결정이라면 원고에 미칠 영향을 읽고 새 목록을 만든 뒤 빌드한다. 해시만 새 값으로 고쳐 내용을 승인한 것처럼 처리하면 순서가 거꾸로 된다.

셋째는 중단 후 재개다. 이번 시험은 운영체제 충돌이나 PDF 쓰기 도중의 강제 종료가 아니다. 별도 시험 코드가 입력 동일성을 확인한 뒤 의도적으로 종료 값 2를 내고, 다음 행동을 ‘새 실행 폴더에서 변환’이라고 기록

했다. 아직 출력은 만들지 않았다. 어디서 멈췄는지 명확하게 남기는 방법을 확인하려고 범위를 작게 정한 시험이다.

재개할 때는 해당 기록을 읽고 같은 입력을 다시 확인한 뒤 새로운 실행 폴더에서 변환했다. 종료 값 0으로 끝났고 생성 PDF의 해시는 정상 실행 결과와 같았다. 같은 정상 입력으로 별도 두 번 만든 PDF·EPUB·통합 Markdown도 세 파일 모두 바이트가 같았다. 이 결과는 선택한 입력과 확인된 로컬 변환 환경에서의 재현성이다. 생성형 AI가 다른 요청에도 같은 답을 한다는 뜻이 아니다.

실제 시험	관찰한 결과
정상 변환	A5 8쪽 PDF와 EPUB·통합 Markdown 생성
그림 제거 사본	종료 1, PDF 미생성
입력 메모 변경·옛 해시 유지	종료 1, PDF 미생성
입력 확인 뒤 의도적 중단	종료 2, 출력 없음·재개 기록 저장
같은 입력으로 새 실행 재개	종료 0, 정상 PDF와 같은 해시
독립 정상 변환 두 번	PDF·EPUB·Markdown 세 파일 바이트 동일

실패 시험은 작업 폴더의 복구 연습이지 실제 고객 자료를 손상시키는 실험이 아니다. 없어진 그림도 수정된 메모도 사본에서만 만들었다. 정상 선택 입력과 이전 출력은 남아 있다. 독자가 시험을 확장할 때도 첫 조건은 같은 보존 경계다. 특히 접근 거절을 시험한다며 실제 권한을 넓히는 행위는 이 연습의 범위에 들어 있지 않다.

지윤의 결정과 도구의 검사를 나눈다

가이드 속 지윤이 받은 메모에는 파일 이름에 역할을 먼저 쓰라고 되어 있다. 새 메모에는 날짜를 먼저 쓰라고 되어 있다. 이것은 가상의 충돌이다. 도구는 두 파일의 내용을 읽고 서로 다르다고 알려 줄 수 있지만 어느 메모가 현재 팀의 승인 규칙인지 스스로 확인할 수 없다. 지윤은 같은 조건을 나란히 적고 담당자에게 한 가지 질문을 보낸다는 계획을 세운다. 이 사례에서 실제 발송을 한 것은 아니다.

답이 없을 때 지윤은 검토본을 준비하고 이름 규칙을 미확정이라고 표시한다. 파일 이름에 승인본이라고 쓰면 완성돼 보이지만 그 단어가 승인 기록을 만들지는 않는다. 그림이나 이름 검사기가 통과했어도 적용 규칙은 대기다. 실습이 짧을수록 ‘한 번에 끝내기’ 욕심으로 이 대기를 지우기 쉬워 더 명확하게 설명했다.

양식 링크가 있는데 파일이 없을 때도 두 판단을 나눈다. 존재 검사는 파일 없음을 보고한다. 지윤은 그 양식이 필수 제출 자료인지 확인한다. 필수라면 기다리고, 설명에 필요 없는 장식이라면 삭제를 선택해 새 사본에서 참조를 고친다. 검사기 PASS를 얻기 위해 모든 링크를 지우는 행동은 성공이 아니다. 독자가 해야 할 행동에 필요한 근거까지 사라질 수 있다.

마지막에는 원고와 PDF뿐 아니라 질문, 실제 수행한 검사, 미수행 사항, 다음 재개 행동이 남는다. 이 네 가지를 포함해야 다른 사람이 결과를 받아 이어 갈 수 있다. ‘모든 검사 완료’라는 한 줄은 편하지만 아직 EPUB 리더를 열어 보지 않았다면 사실과 다르다. 가상의 검토자도 실습의 실제 제작자도 자신의 결정 범위를 부풀리지 않는 것이 이 장의 공통 원칙이다.

내 제작 카드에 실제 파일을 적는다

다음은 이번 실습을 실제 제작한 뒤 작성한 카드다. 가상 조직의 승인 기록이 아니라 교육용 가이드를 만든 로컬 기록이다. 경로는 실습 폴더 기준이며 패키지에 포함되는 선택 파일과 대응한다. 독자에게 중요한 것은 컴퓨터 전체의 긴 개인 경로가 아니라 필요한 파일을 찾을 수 있는 안내다.

제작 항목	실제 작성한 기록
결과와 제외	교육용 가이드 8쪽, 실제 회사 규칙·외부 공개 제외
선택 원고	manuscript/unit_01.md부터 unit_08.md까지 8파일
선택 목록	manifest.json, 원고·그림·도구·글꼴·입력·README 해시
편집 이유	첫 12쪽판의 반복 끝 문장을 정리, 본문 11pt 유지
실제 검사	입력 검사·PDF 구조·8쪽 자체 시각 검토·EPUBCheck
실패와 복구	누락/입력 변경 중단, 입력 확인 뒤 중단·새 실행 재개
재현과 한계	선택 환경의 세 출력력 동일, 새 AI 호출·기기 검수 아님
다음 사람 판단	실제 EPUB 읽기·글꼴 배포 조건·내용 최종 수용

카드를 쓸 때 계획과 수행 기록을 같은 시제로 적지 않는다. ‘검사할 것’은 다음 행동이고 ‘검사했다’는 정확한 파일의 결과와 연결돼야 한다. 아직 원고만 있으면 그 사실을 적는다. 다른 컴퓨터에서 실행하지 않았다면 이식 완료라고 쓰지 않는다. 수행하지 않은 일을 남겨 두는 기록이 다음 작업자를 돕는다.

내가 적을 항목	내용
	만들 결과·독자·제외 행동
	실제 입력과 선택 버전
	원고·도해·양식의 연결
	새 실행 폴더와 출력 파일
	내용·구조·시각 검사 범위
	실패 사본과 중단 지점
	재개 행동·다시 볼 결과
	실제 수행·미수행·사람 결정

이 장의 끝에는 작지만 열린 문서와 그 문서가 만들어진 경로가 있다. 이를 자기 업무로 바꿀 때는 주제와 이름부터 한 조건씩 바꾸자. 바뀐 원고를 검토하고 새 선택 목록으로 만든 뒤 영향을 받은 결과를 다시 확인한다. 이제 14장에서는 파일이 담긴 ZIP을 실제로 다시 풀고 재빌드하면서 ‘나중에 다시 만들 수 있다’는 말을 파일로 확인한다.

14 / 여러 스킬로 결과물 완성하기

검수하고 다시 만들 수 있게 남기기

결과 확인·재개·공유를 어떻게 끝내는가?

이 장에서 남길 것: 재현 기록·실습 패키지·개인 변형 계획

남길 것은 결과만이 아니다.

다음 사람이 입력을 찾아 다시 만들고, 무엇을 확인했는지 이해하고, 미해결을 이어 받을 수 있어야 작은 제작이 끝난다.

먼저, ZIP을 다시 풀어 같은 결과를 만든다

13장에서 만든 여덟 쪽 가이드의 선택 파일을 실제 ZIP으로 묶었다. 원고 여덟 개, 입력 메모, 실행 안내, 도해, 글꼴과 사용권 안내, 스타일, 변환 도구, 선택 목록, 생성 결과와 체크섬 목록을 명시적으로 넣었다. 압축 파일 내부 항목은 체크섬 목록을 포함해 스물두 개다. 제작 폴더 전체를 통째로 압축하지 않았다. 실패 사본, 렌더 이미지, 개인 컴퓨터의 긴 경로가 들어간 실행 기록은 넣지 않았다.

그 ZIP을 다시 열어 목록이 맞는지, 압축 데이터가 손상되지 않았는지, 각 파일의 해시가 원본과 같은지 확인했다. 새 로컬 폴더에 풀고 안내된 변환 명령을 실제 실행했다. 종료 값 0으로 끝났고 PDF·EPUB·통합 Markdown은 13장의 선택 결과와 바이트가 같았다. 이것은 같은 컴퓨터

의 확인된 도구 환경에서 새 폴더로 옮긴 재빌드 시험이다. 다른 운영체제나 다른 독자의 기기에서 시험한 결과는 아니다.

여기까지 하면 ‘다시 만들 수 있다’는 말에 작은 근거가 생긴다. ZIP이 열리는 것만 확인했다면 원고와 그림이 빠져 재빌드가 안 될 수도 있다. PDF만 담았으면 독자가 고칠 원고를 찾을 수 없다. 파일이 모두 있어도 안내의 명령이 개인 경로를 요구하면 다른 폴더에서 멈춘다. 압축, 재열기, 실제 실행은 서로 다른 문제를 발견하는 세 단계다.

하지만 세 출력이 같다는 결과를 ‘새 AI 입력에서도 같은 품질’이라고 확대하지 않는다. 이번 재빌드는 이미 선택한 원고를 결정적인 변환 도구로 처리했다. 제목 스킬이 새로운 사실을 과장하지 않는지, 계획 스킬이 자료 부족을 완료로 표시하지 않는지는 별도의 생성형 실행 시험이다. 이 책의 E01과 E02에는 강화한 의미 기준의 새 실행 검토가 남아 있음을 구분한다.

완료는 한 개의 초록불이 아니다

작업을 끝낼 때 여러 검사를 한 줄의 PASS로 묶고 싶어진다. 그러나 ‘파일이 있음’, ‘파일이 열림’, ‘문장이 맞음’, ‘실제 리더에서 읽힘’, ‘권리를 검토함’, ‘저자가 승인함’은 서로 다른 질문이다. 한 검사가 다른 질문에 답하지 못한다. 프로그램의 종료 값이 0이어도 날짜를 잘못 썼거나 중요한 양식을 삭제했다면 업무 결과가 맞다고 할 수 없다.

검수의 순서를 내용→구조→시각→재현→인계로 생각해 보자. 이것은 이 책의 제작용 구분이며 공식 표준이나 제품의 합격 점수표가 아니다. 내용 검토는 입력과 결과의 뜻을 비교한다. 구조 검사는 필요한 파일, EPUB 연결, 메타데이터를 본다. 시각 검토는 글자의 크기와 표·그림·페이지의 읽는 흐름을 확인한다. 재현은 선택한 입력과 변환 조건을 다시 사

용해 결과를 비교한다. 인계는 다른 사람이 필요한 자료를 찾아 다음 행동을 이해하는지 본다.

실제 EPUB의 작은 화면과 큰 글자 검사는 마지막까지 남길 수 있지만, 남겼다면 미수행이라고 적어야 한다. EPUBCheck가 오류와 경고를 보고 하지 않아도 작은 화면에서 코드가 잘리는지, 표의 읽는 순서가 이해되는지, 내장 글꼴이 리더 설정과 충돌하지 않는지는 별도 질문이다. XML과 ZIP 검사를 리더 평가로 바꾸어 쓰지 않는다. 검사 이름을 자세히 쓰는 것은 책임을 피하는 표현이 아니라 검토 범위를 전달하는 방법이다.

저자 승인도 기술 검사와 나눈다. 이 실습의 가이드가 열리고 다시 만들어졌다고 실제 팀 규정으로 배포할 허락이 생기지 않는다. 내용·표지·권리·독자 경험을 확인하는 결정권자가 결과를 받아야 한다. ‘로컬 검토본’이라는 상태는 부족한 완성 표시가 아니다. 사람이 결정할 것이 남은 문서의 정확한 상태다.

고친 곳보다 영향받는 곳을 본다

원고 한 문장을 바꾸면 어느 검사를 다시 해야 할까? 먼저 그 문장이 빌드 입력인지 확인한다. 독자용 원고라면 출력이 달라질 수 있으므로 새 선택 목록으로 빌드하고 관련 페이지를 다시 읽는다. 입력 메모만 바뀌어도 원고에 반영해야 할 사실이 있는지 판단한다. 해시만 새 값으로 바꾸고 옛 원고를 계속 사용하는 것은 변화가 없었다는 증거가 아니다.

제목이나 저자명을 고치면 영향 범위가 더 크다. 표지, 제목 페이지, PDF 속성, EPUB 메타데이터, 목차와 다운로드 파일 안내를 함께 본다. 그림 파일만 바뀌어도 캡션, 대체 텍스트, 출처 기록, PDF의 삽입 크기, EPUB의 자산이 영향을 받는다. 그림의 색이 바뀌지 않았다는 이유로 글자나 화살표 검토를 생략할 수 없다.

실습 안내 README를 바꾸는 경우도 생각해 보자. PDF 입력에 README가 없으면 본문 PDF를 다시 만들 필요는 없을 수 있다. 그러나 README가 들어간 ZIP은 달라졌으므로 새로 묶고 압축을 다시 열어 링크와 명령을 확인해야 한다. 반대로 이 장의 E04 선택 목록은 실행 안내도 해시 검사 대상으로 포함한다. 따라서 안내를 바꾸면 그 선택 목록도 새 버전으로 기록해야 한다. 실제 입력 범위를 보고 판단해야 하는 이유다.

원고 변경 → 새 입력 · 새 변환

내용 / 메타데이터 / 정확한 출력

도해 변경 → 설명 · 시각 · EPUB

캡션 / alt / 삽입 크기 / 자산 연결

안내 변경 → 새 ZIP · 다시 열기

목록 / 해시 / 링크 / 실제 명령

실제 선택 입력 범위에 따라 돌아간다

그림 14-1. 바꾼 파일의 이름보다 어느 결과의 입력인지 확인한다. 원고·도해는 새 변환과 내용·시각 검토, 안내는 새 패키지와 링크·명령 확인으로 연결된다. 실제 입력 목록에 따라 범위를 정하는 저자의 제작 모델이다.

검토 기록을 재사용할 때도 정확한 파일을 대조한다. 새 PDF의 한 페이지 이미지가 이전에 본 이미지와 해시가 같고 검사 질문도 같다면 그 동일 범위의 시각 검토를 재사용할 수 있다. 그러나 새 PDF 전체를 이전과 같다고 가정할 수는 없다. 달라진 페이지는 다시 펼친다. 어떤 범위를 재사용했고 무엇을 새로 확인했는지 남겨야 과거 기록이 현재 판정을 가리지 않는다.

패키지는 명시한 파일만 넣는다

압축 전에 독자의 필요를 먼저 정한다. 결과물을 읽기만 할 사람에게는 PDF와 EPUB, 읽는 안내가 필요하다. 실습을 수정할 사람에게는 원고, 선택 입력, 필요한 도구, 가상 입력, 빈 양식, 재실행 방법이 필요하다. 저자 검토 자료와 독자 실습 자료를 나누는 것도 좋은 방법이다. 모든 내부 보고가 독자에게 필요한 것은 아니다.

이 실습에서는 명시한 파일 목록으로 패키지를 만들었다. 이를 허용 목록이라고 부른다. 폴더의 모든 파일을 찾아 넣고 위험해 보이는 것만 빼는 방식보다, 넣을 것을 하나씩 선택하는 방식이다. 원본 프로젝트 스킬 전체를 함께 넣지 않았다. 참고한 제작 원칙과 새 교육용 예제를 구별해야 하고, 원본 저장소의 라이선스 확인 범위를 교육용 폴더의 존재로 대신할 수도 없다.

정상 입력과 결과, 안내를 넣되 실패를 설명하는 데 필요 없는 내부 경로는 제외한다. 실제 실패 출력 일부를 교육용으로 제공할 때도 그것이 합성 입력에서 얻은 자료인지 표시한다. 가상 인물의 이름은 실제 고객 정보가 아니라는 설명을 두고, 실제 사람 자료가 있다면 별도 검토한다. 자신의 문서라고 모든 개인정보를 배포할 수 있다고 가정하지 않는다.

체크섬 목록에는 각 파일의 상대 경로와 SHA256 값을 적는다. 압축 파일 자체의 해시는 별도 제작 기록에 남긴다. ZIP 안에 그 ZIP의 자기 해시를 넣으려 하면 내부 값이 바뀔 때 ZIP도 바뀌므로 순환이 생긴다. 내부 파일을 확인할 목록과 외부 ZIP 식별값을 나누는 작은 설계가 재검사를 쉽게 만든다. 체크섬은 내용의 정확성이나 악성 코드 부재를 보증하는 인증이 아니라 동일성 검사다.

압축을 다시 열 때는 파일 개수만 보지 않는다. 목록에 없는 개인 파일이 들어갔는지, 빠진 원고가 있는지, 같은 이름이 두 번 들어갔는지 확인한다.

압축 데이터의 CRC 검사와 각 파일의 해시 대조를 수행한다. Markdown 안내의 상대 링크가 ZIP 안의 실제 파일로 이어지는지도 본다. 안내가 외부 개인 경로를 가리키면 패키지 파일이 모두 있어도 독자가 사용할 수 없다.

새로운 폴더에서 안내대로 따라 한다

재현 시험은 제작자가 기억으로 실행하지 않는 데 의미가 있다. ZIP을 새 폴더에 풀고 README부터 읽는다. 제작 중 사용했던 숨은 개인 설정이나 미리 만들어 둔 결과 폴더에 기대지 않고 필요한 입력과 명령을 찾는다. 이 실습의 변환 도구는 프로젝트 루트를 자기 파일 위치에서 찾고, 선택 목록의 상대 경로를 그 루트 안에서 해석한다. 컴퓨터 전체를 찾는 동작이 아니다.

실제 E04 패키지에는 원고와 자산, 변환 도구 두 파일, 실행 안내가 있다. 새 폴더에서 다음 명령으로 새 실행을 만들었다. 이미 있는 출력 폴더를 덮는 동작이 아니므로 같은 연습을 반복할 때는 이름을 바꾼다. 도구 설치와 권한 변경은 이 명령에 포함되지 않는다.

```
python3 build/build_editorial_book_v3.py \
--manifest manifest.json \
--run-dir builds/new-run
```

명령이 끝나면 종료 값과 생성 파일을 보고 선택 결과와 비교한다. 실제 시험에서는 같은 로컬 Python·글꼴·변환 파일을 사용했고 세 출력이 동일했다. 이 조건을 함께 적어야 한다. 같은 문서가 다른 라이브러리 버전에서 시각적으로 같아도 압축 방식이나 메타데이터 때문에 바이트가 달라질 수 있다. 바이트 동일과 내용·시각 동일을 구분하고, 확인하지 않은 조건을 보장하지 않는다.

다른 컴퓨터로 옮길 때는 운영체제, Python과 필요한 라이브러리, 글꼴, 실행 경로, 읽기·쓰기 권한을 먼저 적는다. AI 스킬 호출까지 시험할 경우 제품과 모델, 입력, 세션, 스킬 선택 버전도 필요하다. 이번 ZIP 재빌드에는 새 AI 호출이 없었다. 명령이 성공했다고 Codex·ChatGPT·Claude 모두에서 스킬 동작을 검증했다고 말하지 않는다.

실패하면 어디서 멈췄는지 분리한다. 입력 파일 검사에서 실패했는지, 라이브러리를 불러오지 못했는지, 변환 중 오류가 났는지, 출력은 생겼지만 검사에서 실패했는지 기록한다. 없는 프로그램을 자동 설치하고 권한을 넓혀 통과시키지 않는다. 해결 후에는 새로운 실행 폴더에서 이어가고 이전 실패를 지우지 않는다. 사람이 할 다음 행동을 짧게 쓰는 편이 긴 오류 출력만 보내는 것보다 도움이 된다.

가상 검토자는 승인 문장을 먼저 고쳤다

이번 **교육용 가상 사례**의 검토자 하린은 안내문의 날짜 한 곳을 고쳤다. PDF를 다시 만들었지만 EPUB은 이전 파일을 남겼다. ZIP 안내에는 ‘모든 결과에 날짜 수정 반영’이라고 썼다. 날짜 수정 자체는 작아 보여도 서로 다른 결과를 함께 전달한 문제가 생긴 것이다. 하린의 실수와 회사는 가상이며 실제 고객 사고 기록이 아니다.

하린은 ZIP을 발송하기 전에 각 결과의 선택 입력을 대조한다. PDF의 기록은 새 원고를 가리키고 EPUB의 기록은 옛 원고를 가리킨다. 먼저 완료 문장을 고쳐 EPUB 재빌드와 검수가 남았다고 적는다. 이후 같은 선택 목록으로 두 형식을 다시 만들고 정확한 파일을 검사한다. 실패 사실을 숨기고 결과만 바꾸면 다음 사람이 어떤 기록을 믿어야 할지 알 수 없다.

또 다른 가상 문제는 없는 제출 양식이다. 이름 검사가 통과하도록 링크를 지우면 패키지는 열리지만 업무는 수행하기 어렵다. 하린은 필수 여부를

확인하고 필요하면 양식이 도착할 때까지 전달을 보류한다. 필수가 아니라면 담당자의 삭제 결정을 기록하고 원고·도해·출력·안내에서 참조를 함께 제거한다. ‘검사 통과’가 사람의 필요 판단을 대신하지 않는다.

마지막으로 하린은 스킬의 지시 문장도 고쳤다. 계획 상태를 표시할 때 자료가 없으면 대기로 적으라는 수정이다. 그런데 PDF 재빌드만 성공했다고 그 스킬의 새 행동까지 검증한 것은 아니다. 새로운 입력에서 실제 호출 시험이 필요하다. 시험을 하지 못한 상태에서는 수정안과 기존 실제 시험 결과를 나누어 전달한다. 이 책의 E02 보완안도 같은 원칙으로 읽는다.

이 사례의 끝은 완성이라는 단어 하나가 아니라 서로 맞는 파일과 정확한 남은 일이다. 하린은 수용할 원고와 전달 범위를 사람이 결정하도록 남긴다. 기술 검수가 가능하다고 즉시 공개하는 것은 별개다. 자기 업무에서도 먼저 이 경계를 정하면 작은 변경 뒤 무엇을 다시 볼지 훨씬 명확해진다.

다음 사람이 읽을 재현 카드를 남긴다

아래 카드는 E04 실습 ZIP을 실제 다시 열고 재빌드한 뒤 작성했다. 가상의 팀 승인이나 전체 전자책 출판 완료 기록은 아니다. 같은 컴퓨터에서 새 폴더로 옮긴 시험이라는 조건을 카드 안에서도 유지한다.

기록할 질문	실제 E04 패키지 기록
어떤 파일인가	교육용 가이드 선택 원고·도구·결과의 ZIP
무엇을 넣었는가	명시 22항목, 내부 체크섬 포함·원본 프로젝트 제외
다시 열었는가	ZIP 목록·CRC·모든 항목 해시 대조
어디서 실행했는가	같은 로컬 환경의 새 폴더, 안내된 명령
무엇이 같았는가	PDF·EPUB·통합 Markdown 세 파일 바이트 동일
무엇을 시험하지 않았는가	새 AI 호출·타 OS·실제 EPUB 리더
무엇이 남는가	사람 수용·권리 최종 검토·자기 업무 조건 확인
다음 변경의 원칙	새 선택 입력·새 실행·영향받는 검사·새 ZIP

이 카드에는 모든 긴 로그를 옮길 필요가 없다. 결과 파일과 정확한 보고를 찾을 수 있도록 연결하고, 미수행과 다음 행동을 알기 쉽게 적는다. 개인 경로나 비밀값을 공유 안내에 넣지 않는다. 독자가 자신의 컴퓨터에서 찾아야 할 상대 위치를 제공한다. 이름이 같아도 파일 내용이 바뀌면 새 카드와 패키지 검사가 필요하다.

내가 적을 항목	내용
선택 결과·원고·버전	
패키지에 넣을 명시 목록	
필요한 도구·환경·권한	
실제 내용·구조·시각 검사	
재열기·해시·링크 확인	
새 폴더 실행 결과와 한계	
미수행·사람 결정·재개 위치	
다음 변경과 재검사 범위	

스킬을 배포할 때도 폴더의 존재와 제품의 실제 검색·호출을 나눈다. 스킬은 필요한 지시와 자료를 연결하는 형식이며 사용 환경에 따라 읽는 위치와 호출 방식이 다르다. 실제 적용 전에는 사용 제품의 현재 공식 안내와 허용 범위를 확인하자. 이번 실습 ZIP은 자동 설치 프로그램이 아니다. [OpenAI Build skills](#), 2026-10-03 확인.

처음의 질문으로 돌아가 보자. 반복 설명을 줄이려면 더 긴 프롬프트만 필요한 것이 아니었다. 어떤 입력으로 무엇을 만들고, 어디서 멈추고, 무엇을 확인하고, 다음 사람이 무엇을 이어 할지 정해야 했다. 이제 자기 업무 하나를 골라 작은 스킬과 결과를 만들 수 있다. 성공을 크게 말하기보다 고칠 위치와 다시 시작할 파일이 남는 작업을 선택하자. 그 작은 기록이 다음 반복에서 실제로 쓸 수 있는 자산이 된다.

REFERENCE / 다시 찾는 안내

작은 결과를 다음 반복으로 넘기기

한 번 쓰고 버릴 요청과 다시 쓸 작업 방법 사이에는 정리가 필요하다. 입력이 달라져도 유지할 기준, 자료를 기다릴 조건, 사람이 말을 결정을 적는 것이다. 이 책은 그 정리를 세 스킬과 작은 가이드에 적용했다.

제목에서는 근거 없는 성과를 확정하지 않았고, 계획에서는 제외와 선행 조건을 적었다. 안내문에서는 사실·판단·표현을 나눴고, 가이드에서는 선택 입력·새 실행·출력 검사·패키지를 연결했다. 자기 업무에는 결과의 모양보다 그 이유를 가져가자.

반복 업무 하나에서 작은 조건부터 정한다. 정상 입력과 실패 입력을 읽고 어디서 멈출지 적는다. 가상 규칙을 실제 팀의 사실로 가져오지 않고 자신의 자료와 승인자를 확인한다. 모르는 것을 질문으로 남기는 태도가 실무적인 사용법이다.

결과를 받은 뒤에도 선택 원고, 시험 조건, 수정 이유, 마지막 파일과 다음 행동을 남긴다. 수행하지 않은 검사를 정확히 표시한 기록은 안전하게 이어 갈 발판이다.

이 로컬 판에도 최종 원고·표지 수용, 실제 EPUB 읽기와 권리 판단이 남는다. 그 결정을 숨기지 않는 것까지 제작의 일부다. 자신이 매번 다시 설명하는 일 하나를 다음 사람이 이어 쓸 작은 작업으로 바꿔 보자.

REFERENCE / 다시 찾는 안내

내가 쓰는 말로 바꾸기 · 파일

SKILL.md — 다시 쓸 작업 설명서

스킬의 핵심 지시 파일이다. 시작 표지에는 이름과 언제 사용할지 설명을 적고, 본문에는 입력·진행·결과·중단 경계를 쓴다. 파일 이름만 같다고 내용의 품질이 같거나 제품에서 자동으로 실행되는 것은 아니다.

YAML 표지 — 파일의 첫 소개

두 줄의 --- 사이에 적는 이름·설명 같은 항목이다. 본문과 다르게 프로그램이 읽을 형태를 지켜야 한다. 괄호와 따옴표를 많이 외우기보다 실제 파싱과 원본 대조로 값이 제대로 읽히는지 확인한다. 03장에서 작은 표지를 만들고 07장에서 폴더 이름과 연결한다.

참고 자료와 양식 — 읽을 때와 쓸 때의 보조

참고 자료는 판단에 필요한 설명이며 양식은 결과를 정리할 빈 틀이다. 긴 내용이라는 이유만으로 모두 본문 밖에 빠지 않는다. 언제 읽어야 하는지 지시에 연결하고, 현재 입력과 충돌하면 어느 규칙이 우선인지 확인한다.

상대 경로 — 이 폴더부터 찾는 주소

개인 컴퓨터 전체 주소 대신 지정한 폴더를 기준으로 파일을 찾는 표현이다. 다른 폴더로 옮겨도 관계를 유지하기 쉽지만, 어디를 기준으로 읽는지 안내해야 한다. 공백과 한국어가 있는 경로는 명령의 따옴표 처리도 확인한다.

선택 목록 — 실제 사용할 파일 명단

원고·그림·도구·글꼴의 경로와 해시를 명시한 목록이다. 폴더에서 최신 파일을 추측하는 방법이 아니다. 목록을 고쳤다고 내용이 승인되는 것은 아니므로 선택 전에 사람의 내용 검토가 필요하다.

해시 — 내용이 같은지 보는 값

파일에서 계산한 식별값이다. 이름이 같아도 내용이 다르면 값이 달라질 수 있다. 입력과 출력의 동일성을 대조하는 데 사용한다. 진실성·보안성·사용권을 인증하는 만능 증명이 아니다.

REFERENCE / 다시 찾는 안내

내가 쓰는 말로 바꾸기 · 상태

합성 시험 — 만든 입력으로 실제 실행

비민감 연습 입력을 작성하고 실제 AI나 도구에 넣어 관찰하는 시험이다. 입력은 교육용으로 만든 것이지만 실행을 실제 수행했다면 그 출력은 관찰 자료다. 실사용자 평가나 전체 서비스 성능 통계와 구분한다.

가상 사례 — 판단을 설명하는 이야기

저자가 만든 인물·상황·수치로 원리와 결정 과정을 보여 준다. 실제 출력처럼 꾸미지 않는다. 사례의 결과가 좋아도 제품의 실증 효과를 입증하지 않는다. 필요한 조건을 이해한 뒤 자기 자료로 확인한다.

종료 값 — 프로그램이 끝난 표시

이 책의 작은 도구는 정상 처리를 0, 지정 문제 발견을 1처럼 구분한다. 모든 도구가 같은 의미를 쓰는 것은 아니다. 0이 나왔다고 업무 내용·시각·권리·승인이 모두 맞다는 뜻은 아니다. 무엇을 검사한 도구인지 함께 읽는다.

자체 검토 — 제작자가 다시 읽은 확인

원고를 저장한 뒤 전체 재독하고 입력·판단·행동·실패·사람 결정을 대조한 기록이다. 독립 전문가나 실제 입문 독자의 수용과 다르다. 파일 존재와도 다르다. 검토한 정확한 버전과 범위를 남긴다.

재현 — 같은 조건으로 다시 확인

선택한 입력과 도구를 써서 새 실행을 만들고 결과를 비교하는 것이다. 결정적인 문서 변환의 바이트 동일과 생성형 AI의 새 입력 대응은 다른 시험이다. 환경을 기록하지 않은 재현성 주장은 범위를 알기 어렵다.

보류와 승인 — 아직 할 수 없는 일과 사람 결정

자료·권한·결정이 없으면 결과를 준비했더라도 보류할 행동이 남는다. 승인이라는 단어를 파일 이름에 넣어도 실제 결정 기록은 생기지 않는다. 초고 저장, 검사 통과, 저자 수용, 외부 전달을 각각 적는다.

REFERENCE / 다시 찾는 안내

환경이 다르면 먼저 볼 것

스킬 형식과 제품 사용법을 나눈다

SKILL.md와 참고 자료를 구성하는 원리는 여러 환경에서 이해할 수 있지만 검색 위치·호출 방식·지원 범위는 제품과 버전에 따라 확인해야 한다. 다른 제품에서 이름이 같다고 모든 동작이 같지는 않다. 설치 전에 공식 안내와 현재 설정을 읽는다. 이 묶음은 사용자 전역 폴더로 자동 복사하지 않는다.

공식 자료에서 Codex의 명시 호출과 설명에 따른 선택을 확인할 수 있다. 그러나 설명을 적었다고 매번 기대대로 선택된다는 실험 결과는 아니다. 자신의 입력과 선택 파일을 기록하고 실제 읽기·호출을 관찰한다.

[OpenAI Build skills](#), 2026-10-03 확인.

확인된 로컬 환경을 참고한다

이 제작에서 새 파일 검사와 문서 변환에 사용한 Python은 3.12.14이며 ReportLab 4.4.9와 pypdf를 사용했다. EPUBCheck 5.3.0은 OpenJDK 17의 명시 경로로 실행했다. 이 버전이 모든 독자의 필수 버전이나 최신 권장 버전이라는 뜻은 아니다. 실행한 환경을 식별하기 위한 기록이다.

실제 합성 스킬 시험은 지정한 Codex CLI·모델·격리 세션에서 수행했다. 현재 로컬 파일 재검사는 새 AI 시험이 아니다. Windows나 다른 제품

의 행동 검증은 미수행이다. 자기 환경에서 반복할 때는 모델과 버전, 스킬 파일, 입력과 기대 조건을 함께 남긴다.

비용과 권한은 따로 묻는다

파일을 읽는 동작, 새 파일을 쓰는 동작, 명령을 실행하는 동작, 외부 서비스에 보내는 동작을 구분한다. 계정 설정이나 파일 존재가 외부 사용 승인으로 자동 이어지지 않는다. 비용 조건은 현재 계정과 공식 안내를 확인한다. 이 책은 특정 요금이나 무제한 실행을 약속하지 않는다.

다른 컴퓨터로 옮겼을 때 첫 목표는 작고 비민감한 정상 입력 하나다. 자료가 읽히고 지정 폴더에 새 결과가 생기는지 확인한다. 이후 누락과 충돌, 접근 실패를 사본에서 시험한다. 기존 실제 자료를 지우거나 전역 설정을 바꾸어 시험을 통과시키지 않는다.

REFERENCE / 다시 찾는 안내

작은 실패 뒤 돌아가는 길

파일이 없다는 말부터 분류한다

입력 자체가 없는지, 경로가 다른지, 접근이 허용되지 않는지 먼저 나눈다. 제공되지 않은 파일은 요청하고, 상대 위치가 잘못됐다면 안내와 목록을 고친다. 접근 거절은 경로 오타와 다르다. 다른 폴더나 도구로 우회해 읽지 않고 사람이 허용 범위를 정하도록 한다.

그림이 빠졌다면 업무상 필요한 설명인지 편집자가 판단한다. 필요하다면 복구를 기다린다. 삭제할 수 있는 장식이라면 결정 후 본문·캡션·alt·선택 목록에서 참조를 함께 제거한다. 검사기가 파일 없음을 보고했다고 삭제 필요성까지 결정한 것은 아니다.

출력은 있는데 완료 기록이 없다면

파일이 열리는지와 정확한 입력을 사용했는지 본다. 중간 결과를 가장 최신이라는 이유로 선택하지 않는다. 이전 검증본은 보존하고 새 실행 폴더에서 이어 간다. 마지막 완료 단계와 미완료 행동을 짧게 남긴다.

E04의 중단 연습은 입력 확인 뒤 의도적으로 멈춘 것이다. 운영체제 충돌이나 쓰기 중 강제 종료에서 복구를 시험한 것은 아니다. 자신의 작업에서 그 문제가 생겼다면 해당 파일을 보존하고 별도 복구 조건을 확인해야 한다. 작은 연습 범위의 결과를 다른 장애에 확대하지 않는다.

검사가 통과했는데 내용이 틀리다면

프로그램이 검사하는 조건을 읽는다. E03 기본 검사기는 파일·기본 태그·일부 참조를 보며 날짜의 사실이나 모든 HTML 중첩, 실제 시각 경험을 판정하지 않는다. 입력과 출력의 의미 대조를 따로 수행한다. 검사를 맞추려고 필요한 정보를 삭제하지 않는다.

수정은 한 이유씩 진행한다. 바꾼 입력과 지시, 기대 조건을 기록하고 원래 정상 입력이 여전히 가능한지도 본다. 생성형 행동을 바꾼 수정은 새 AI 시험이 필요하다. 새 호출의 비용이나 권한이 확인되지 않으면 수정안과 미수행 기록을 나누어 남긴다.

다음 사람이 이어 할 문장을 남긴다

‘실패함’ 대신 ‘선택 목록의 그림 파일이 없어 변환 전 중단, 원본 보존, 허용된 그림 사본을 받은 뒤 새 실행’처럼 적는다. 실제 한 검사와 아직 못 한 검사를 구분한다. 승인되지 않은 외부 행동은 다음 행동의 조건으로 남긴다.

REFERENCE / 다시 찾는 안내

공식 자료와 교육용 예제

제품 사실의 기준

[OpenAI Build skills](#)는 스킬 형식, 지시·자료·선택 스크립트, 읽기와 호출 방식을 확인할 기준이다. 이 책의 장별 확인일은 2026-10-03이다. 제품 공식 기능과 저자의 제작 역할 분류를 같은 표준으로 설명하지 않는다.

[Agent Skills specification](#)은 파일 형식의 공개 기준을 읽을 자료다. [YAML 1.2.2](#)는 표지의 값과 줄 접기 표현을 확인할 자료다. 둘 다 책의 가상 업무 승인 규칙을 정의하는 문서가 아니다.

[Python HTMLParser](#)와 [pathlib](#)는 작은 HTML 검사기와 경로 처리의 기능·한계를 확인하는 1차 자료다. [zsh Shell Grammar](#)는 명령의 줄 연결과 따옴표 설명을 확인할 자료다. 해당 주장은 본문 가까이의 확인일과 범위를 우선 읽는다.

[MDN HTML basics](#)는 문서 구조와 표현의 기초를 읽는 자료다. 전체 웹 개발을 이 책의 범위로 확대하지 않는다. 실행 보안의 제품별 범위는 [Codex sandboxing](#)의 현재 안내와 실제 허용 설정을 확인한다.

원본 프로젝트를 참조하는 방식

Coreline-AI의 제목·개발 계획·학습 자료 스킬에서 작업 경계와 입력·출력, 실패 처리의 원리를 참고했다. 플랫폼별 제목 규칙, 전체 개발 계획 워크플로, 다중 HTML 모드와 원본 자산을 그대로 독자 패키지에 복제하지 않았다. 교육용 E01~E03은 별도의 작은 예제로 새로 썼다.

원본 저장소의 라이선스가 확인되지 않은 자료를 폴더가 있다는 이유로 자유 배포로 판단하지 않는다. 참고 원리, 실제 선택 파일, 인용 범위, 교육용 제작물의 출처 기록을 나눠 관리한다. 이 책의 출처 목록은 법률상 독창성이나 비침해 보증이 아니다.

실제 기록을 읽는 범위

기존 합성 시험은 당시 지정 기대 조건에서 관찰한 결과다. 이후 원고를 쓰며 강화한 의미 기준의 미해결을 발견하면 기존 통과를 숨기거나 전체 통과로 확대하지 않는다. 새 실행 없는 수정 제안은 그 상태로 표시한다. 실제 제품 화면이 없는 도해도 설명용 기록으로 읽는다.

REFERENCE / 다시 찾는 안내

실습 묶음의 파일 안내

세 스킬과 한 결과물을 찾는다

E01-title은 과장 없는 제목 후보를 만드는 연습이다. 시작본과 역사적 선택 개정본, 입력 양식, 좋은/나쁜 예시, 시험 카드가 있다. 강화한 성공·실제 수식어 기준의 새 AI 시험은 남아 있으므로 실제 게시 제목을 자동 확정하지 않는다.

E02-plan은 목표·포함/제외·선행조건·결과·완료 기준을 다룬다. 역사적 선택본과 자료·양식 분리 제안은 같은 검증 상태가 아니다. 분리 제안의 파일 존재와 YAML 표지를 정적으로 확인했지만 새 행동 시험을 했다고 쓰지 않는다. 자료 부족에서 완료 상태를 확인하는 미해결을 함께 읽는다.

E03-handout은 메모를 단일 HTML 안내문으로 표현하는 연습이다. 기본 검사기는 별도 Python 파일이다. 실제 합성 정상 출력, 실패 출력의 사본과 로컬 수정 사본을 구분한다. 출력 날짜나 연락처는 제공된 입력과 대조해야 한다. 코드 검사 통과만으로 외부 발송하지 않는다.

E04-guide는 가상 팀 검토 파일 준비 가이드를 원고부터 실제 PDF·EPUB·Markdown으로 만든 묶음이다. 선택 목록과 변환 도구, 글꼴·OFL 안내, 결과를 포함한다. 원고 여덟 파일을 사용하며 가이드 PDF는 선택 판에서 여덟 쪽이다. 일곱 제작 역할을 별도 에이전트로 모두 호출한 증거는 아니다.

다시 실행할 때

각 폴더의 README를 먼저 읽는다. 새로운 연습 사본을 만들고 필요한 도구가 있는지 확인한다. E04에서는 폴더 안에서 선택 목록을 명시하고 없는 실행 폴더 이름을 사용한다. 결과와 기록을 다시 열어 정확한 입력과 연결한다. 파일 묶음의 README와 체크섬 목록이 현재 안내다.

독자의 전역 스킬 폴더로 자동 설치하지 않는다. 사용 제품의 검색 위치와 스킬 호출 방식은 공식 안내와 허용 범위를 확인한 뒤 적용한다. 로컬 파일 읽기·명령 실행·새 AI 호출·외부 공개는 같은 행동이 아니다. 이 묶음은 공개·판매 승인도 대행하지 않는다.

원본 프로젝트 전체나 내부 전체 로그를 담지 않는다. 가상 입력과 교육용 파일로 연습한다. 자신의 조건을 추가할 때는 무엇이 실제이고 무엇이 예시인지 계속 표시한다.

새 입력으로 다시 시험하기

기존 정상 입력을 기준으로 남긴다

새 스킬을 고칠 때 처음 정상 입력 하나를 남겨 둔다. 실패를 고치다 원래 필요했던 결과가 사라졌는지 확인할 기준이다. 기대 조건은 실제로 관찰 가능한 행동으로 적는다. ‘ 좋게 작성 ’보다 ‘ 제공되지 않은 날짜는 질문으로 남기고 출력에 확정하지 않음 ’처럼 쓰면 입력과 결과를 대조할 수 있다.

E01을 자기 업무에 적용할 때는 숫자뿐 아니라 성공·실패·검증 같은 수식어가 입력 근거를 넘지 않는지 본다. 본문이 아닌 메모만 받았다면 본문 검토를 했다고 적지 않는다. 적절한 제목 후보와 사람의 최종 선택을 구분한다.

E02에서는 자료가 없을 때 목표를 추정하거나 계획 상태를 완성이라고 표시하지 않는지 본다. 사람이 나중에 정할 단계가 남았다면 조건을 적고 보류한다. 계획 표현이 만들어진 것과 실행이 허용된 것을 같은 값으로 묶지 않는다. 분리 제안의 지시를 고친 경우에도 새 실제 호출로 확인하기 전에는 행동 검증 완료가 아니다.

E03에서는 메모 충돌, 그림 누락, 경로 미접근을 각각 시험한다. 검사기를 통과시키려 필수 자료를 삭제하지 않는다. HTML 구조 검사와 사실 대조, 실제 리더·화면 검토를 나누고 수행한 범위만 기록한다.

한 가지를 바꾸고 다시 읽는다

독자가 만든 시험 숫자를 성능 통계로 발표하지 않는다. 입력·세션·모델·스킬 버전·기대 조건·실제 결과를 기록한 작은 검토다. 다른 사람의 자료를 쓰거나 비용이 드는 호출을 할 때는 범위와 권한을 먼저 확인한다.

실패하면 입력·지시·자료·도구 중 고칠 곳을 적고 새 사본을 만든다. 다음 시험에서 무엇이 달라질 것으로 기대하는지 먼저 정한다. 결과를 보고 기대 조건을 몰래 완화하지 않는다. 개선이 확인되면 원래 정상 입력과 다른 실패 입력도 다시 읽는다.

새 입력 시험이 불가능하면 수정안, 기존 실행 결과, 새 시험 미수행을 따로 남긴다. 정확한 한계가 있는 기록은 다음 사람의 출발점이 된다. 스킬은 한 번의 완성 선언보다 이렇게 반복해서 고칠 수 있을 때 실무 자산에 가까워진다.