AI를 잘 쓰는 사람은 무엇이 다른가

한국기초과학지원연구원 | AI 활용법

구요한 / CMDSPACE

2026.09.10 목요일 14:00–15:00

제가 소개할 것은
제가 운영하는 구조입니다

01

10,000+ 노트의 지식 생태계

02

기업 경영진 교육과 코칭 / 연구자 교육

03

지식관리 → 소프트웨어 → 강의와 글

04

cmdspace.work

AI는 쓰는데,
일은 왜 매번 다시 시작할까요?

질문은 많아졌다 /	**업무 기록은 남았는가**
답은 빨라졌다 /	**우리 맥락은 이어지는가**
초안은 늘었다 /	**확인할 일도 늘지 않았는가**

잘 쓰는 사람은
세 가지를 다르게 설계합니다

맡길 업무

기억할 맥락

통과할 기준

묻지 않고 맡긴다

답변 한 번이 아니라

반복될 업무를 넘긴다

좋은 질문보다
작업 계약이 먼저입니다

질문

질문: 오늘 한 일 정리해줘

작업 계약

위임: 매일 정해진 시각에 승인된 기록을 읽어라

산출: 결정 / 근거 / 다음 행동을 구분해 저장하라

예외: 근거가 없으면 추정하지 말고 비워라

업무일지는 기억을 짜내는 문서가 아니라 기록을 모으는 결과물입니다

2026.09.08 실제 자동 생성 업무기록 발췌

19:13 … 시스템 문서 갱신

cmds-share: … 표가 raw 파이프 텍스트로 공유되던 버그 수정

활동 시각 + 한 일 + 구체적 변경 내용

연구와 행정은 달라도 기록의 구조는 같습니다

연구

연구: 관찰 / 조건 / 해석 / 다음 실험

행정

행정: 요청 / 근거 규정 / 결정 / 다음 조치

공통: 원문 링크 / 담당 / 확인할 사항

자동으로 쓰는 것보다
무엇을 읽게 할지가 중요합니다

01

읽기 범위는 최소화

02

원본은 보존

03

사실과 추정은 분리

04

외부 전송은 승인된 경로만

여러분의 금요일 오후에서 반복되는 일 하나만 고른다면?

언제 시작할까?

무엇을 읽을까?

어디에 남길까?

무엇이면 멈출까?

매번 설명하지 않는다

AI의 기억력을 기대하기보다

맥락이 남을 자리를 만든다

맥락은 길게 붙이는 것이 아니라 필요한 만큼 찾아 쓰는 것입니다

업무 배경:

목적 / 대상 / 제약

참고 근거:

원문 / 날짜 / 버전

이전 판단:

결정 / 이유 / 남은 질문

포맷은 사고를 강제합니다

내용만 남길 때

‘회의 내용’만 있으면 요약이 남습니다

구조를 남길 때

‘결정 / 근거 / 담당 / 기한’이 있으면 일이 남습니다

마크다운은 목적이 아니라

사람과 AI가 함께 읽는 연결 형식입니다

좋은 검색은 답에서 끝나지 않고 근거로 돌아갑니다

질문

질문: 반복 업무를 맡기기 전에 무엇을 정해야 할까?

찾은 노트

찾은 노트: Human-Agent Handoff Cycle

원문 확인

원문 발췌: 에이전트가 만들고,
사람이 판단하고,

다시 에이전트에게 넘긴다

실제 검색·원문 확인 | 2026.09.10

쌓아둔 자료와
다시 쓸 지식은 다릅니다

01

Connect:

필요한 것을 발견하고 연결

02

Merge:

여러 근거를 내 언어로 통합

03

Develop:

업무와 도구에 적용

04

Share:

결과를 내고 다음 맥락으로 환류

새 담당자와 새 AI에게 같은 설명을 다시 하고 있습니까?

맥락 파일 하나

원문으로 가는 링크

누가 언제 고치는지

검수하지 않고 기준을 준다

검증을 없애는 것이 아니라

기준을 먼저 실행 가능하게 만든다

1,000개 가상 응답 실행을 하나의 규칙으로 관리합니다

정책연구 프로젝트의 합성 페르소나 사례

자동화한 것:

생성 / 검사 / 실패 회수

실제 사람 대상 조사나 대표 여론이 아닙니다

규모가 커지면
좋은 요청만으로는 부족합니다

입력:

페르소나 / 문항 / 실행 조건

출력:

정해진 항목 / 유효한 값 / 실행 식별자

실패:

누락 / 형식 오류 / 중단을 따로 기록

기준을 주면
실패도 다음 행동이 됩니다

01

기준:

점수는 1–5 / 근거는 필수

02

입력:

점수 7, 근거 없음 → 검사 실패

03

다음 행동:

오류를 돌려주고 재시도

04

범위·근거를 다시 검사 → 통과 또는 중단

교육용 예시 | 실제 조사 문항·응답 아님

진행률 하나로는 끝났는지 알 수 없습니다

계획된 실행 / 시도한 실행

검사를 통과한 결과 / 재시도 대기

남은 실패 / 사람이 판단할 예외

운영 원리 예시 | 비공개 연구 결과는 미포함

형식이 맞는 답과 믿을 만한 결론은 다릅니다

자동 확인:

누락 / 값의 범위 / 중복 / 형식

사람의 판단:

대표성 / 편향 / 해석 / 활용 범위

‘통과했다’와 ‘타당하다’를 구분한다

세 가지 차이는
결국 하나의 구조입니다

업무를 정의하고

맥락을 남기고

기준으로 반복한다

같은 주간보고도 맡기는 방식이 달라집니다

01

Prompt:

이번 주 보고서 써줘

02

Context:

활동 기록과 지난 보고서를 함께 제공

03

Harness:

승인된 폴더를 읽고 초안만 저장

04

Loop:

누락을 검사하고 예외만 담당자에게

세컨드 브레인은
보관함을 넘어 작업 환경이 됩니다

노트: 관찰과 출처가 남는다

위키: 반복 쓰는 개념과 관계를 엮는다

에이전트: 필요한 지식을 읽어 산출물을 만든다

새 결과와 검토 기록 → 다음 맥락

AI를 잘 쓰는 능력은
이미 하던 일을 다시 설계하는 능력입니다

연구자: 재현 가능한 조건과 근거

행정·지원자: 반복 가능한 절차와 책임

공통: 좋은 결과의 기준을 설명할 수 있는가

내일부터는
파일 하나와 업무 하나로 시작하세요

입력과 작업 기준

입력: '금요일 공개 세미나 안내, 장소 미정'

요청: 제공된 문장만 사용해 공지 초안을 써라

형식: 일정 / 장소 / 확인할 일

기준: 모르는 정보는 '미정'으로 남겨라

결과

결과: 금요일 / 미정 / 장소 확정 필요

교육용 예시 | 승인된 AI 작업 공간에 자료와 기준을 함께

자기 시스템은
말이 아니라 근거로 점검합니다

Prompt:	요청
Context:	맥락
Harness:	실행 환경
Loop:	반복과 검증
Interop & Governance:	연결과 책임

AKM Index | akm.cmdspace.work

빈손으로 시작하지 않도록 만들어 둔 것을 드립니다

시스템 파일 | system.cmdspace.work

CMDS 스타터킷 | github.com/johnfkoo951/cmds-vault

LLM Wiki 스타터킷 | github.com/johnfkoo951/cmds-llm-wiki

CmdMD | cmdmd.cmdspace.work

AKM Index | akm.cmdspace.work

좋은 답변은 한 번을 돕고,
좋은 구조는 다음 일도 돕습니다

01

묻기에서 맡기기로

02

재설명에서 맥락으로

03

사후 수정에서 기준으로

구요한 / CMDSPACE

필요한 부분만
여기서 꺼내 쓰세요

원리와 포맷 | S32–S41

에이전트와 직접 만든 도구 | S42–S47

판단과 검증 | S48–S52

업무 적용과 다음 단계 | S53–S64

v1에서 v4로 갈수록 사람이 매번 챙길 일이 바뀝니다

01

v1 Prompt:

요청을 만든다

02

v2 Context:

배경을 제공한다

03

v3 Harness:

환경을 설계한다

04

v4 Loop:

검증과 반복을 운영한다

프롬프트에도
업무의 최소 단위가 있습니다

목적 / 대상 / 제공 근거

원하는 결과 / 제약 / 모를 때의 처리

나쁜 요청: 멋지게 만들어줘

좋은 요청: 근거 없는 주장은 분리해줘

맥락 파일은
업무의 안내서 한 장입니다

01

목적과 독자

02

현재 상태와 결정 이력

03

참고 원문과 기준 버전

04

금지 행동과 최종 승인자

보여주는 형식과
함께 일하는 형식은 다릅니다

전달용:

PDF / PPTX / DOCX / 엑셀 보고서

교환 가능한 작업용:

Markdown / JSON / YAML / CSV

원천:

원문 / 로그 / 전사 / 계측 데이터

원본은 버리지 않는다

내용만큼
‘이 자료가 무엇인가’도 남겨야 합니다

01

유형 / 설명 / 작성 주체

02

생성일 / 수정일 / 별칭 / 태그

03

기관 업무에는 권한·보존 기준도 추가

04

메타데이터는 검색과 관리의 단서입니다

폴더는 위치를 알려주고, 링크는 관계를 알려줍니다

폴더: 이 자료를 어디에 둘까?

링크: 어떤 판단과 연결될까?

메타데이터: 무엇이며 언제 만든 자료인가?

셋을 함께 설계한다

자료를 쌓기 전에
다루는 질서를 설계합니다

원문:

무엇을 보존할까

위키:

어떻게 연결하고 갱신할까

스키마:

무엇을 읽고, 어떻게
처리하고, 무엇을 남길까

Schema는 Harness다

LLM Wiki는
요약 파일을 쌓는 곳이 아닙니다

원천 자료:

보존한다

위키:

개념을 연결하고 갱신한다

질의:

근거로 답하고 빈 곳을 찾는다

한 번 쓰고
여러 형태로 내보냅니다

마크다운 원본 + 메타데이터

→ PDF / HWPX / 웹 / AI 참고자료

모든 포맷을 대체하는 것이 아니라

모든 포맷이 시작되는 포맷

Markdown Is All You Need

찾아오는 일과
미리 엮어두는 일은 함께 갑니다

검색: 지금 질문에 맞는 원문을 찾는다

컴파일: 반복 쓰는 개념과 관계를 정리한다

필요한 근거는 언제든 원문으로 돌아간다

규칙, 스킬, 연결 도구는 서로 다른 일을 합니다

규칙:

지켜야 할 경계

스킬:

반복 작업의 수행 절차

MCP:

외부 도구·자료에 연결하는 규약

이름보다 실제 권한을 확인합니다

에이전트를 늘리기 전에 역할과 검증을 나눕니다

조사 담당 → 작성 담당 → 별도 검증 담당

공통 기준 / 책임 있는 통합자

같은 파일을 동시에 덮어쓰지 않기

9Yohan은
역할을 명시한 운영 실험입니다

01

연구 / 글쓰기 / 교육

02

창작 / 분석 / 관계

03

개발 / 커뮤니티 / 자문

04

9yohan.cmdspace.work

OmniControl은
여러 실행을 놓치지 않기 위한 도구입니다

01

어떤 작업이 살아 있는가

02

무엇이 끝났고 무엇이 막혔는가

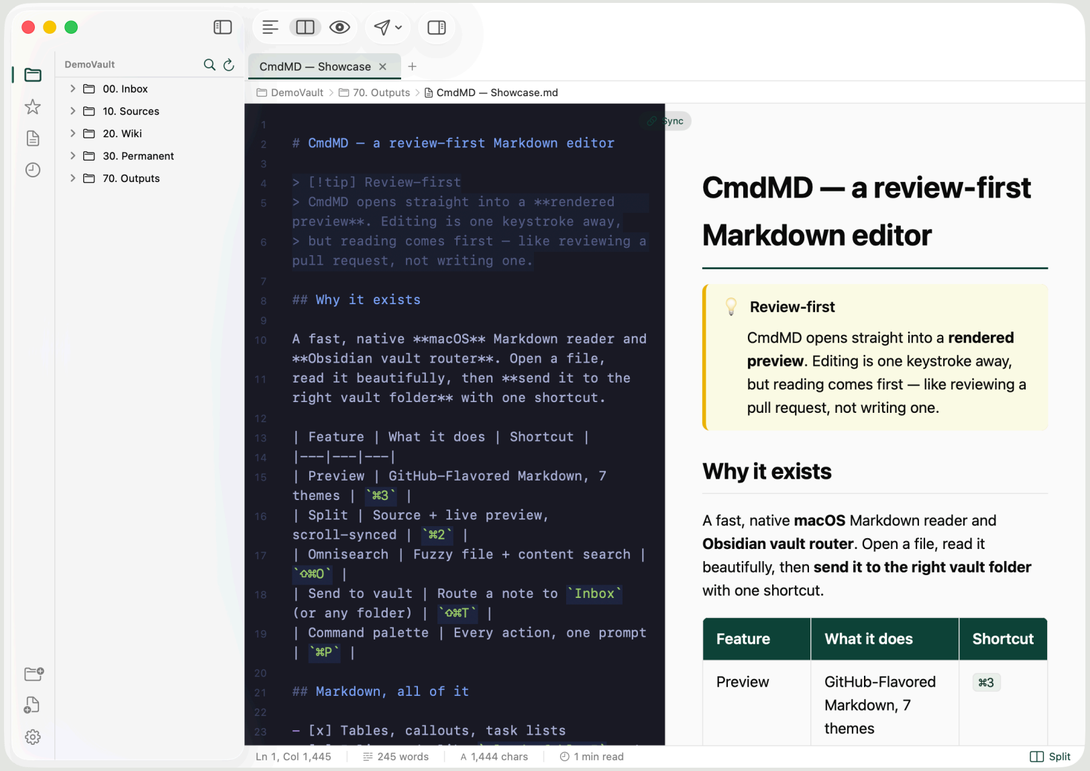
03

어느 기록으로 돌아가야 하는가

04

자동화에도 관측 가능한 상태가 필요합니다

CmdMD는
마크다운을 읽는 데서 출발합니다



렌더된 문서로 먼저 검토

필요할 때 소스 편집

지식 저장소로 보내기

cmdmd.cmdspace.work

개인 지식이
질문의 출발점이 될 수 있습니다

Gobi:

개인 지식과 대화를 연결하는 제품 맥락

중요한 것은 제품명보다

어떤 지식을 누구에게 보여줄 것인가

위임과 판단 포기는 다릅니다

위임: 목적과 기준을 주고 예외를 받는다

판단 포기: 답이 나왔으니 그대로 믿는다

최종 책임은 도구 밖에 남습니다

권한은
업무별로 잘라서 줍니다

읽기:

무엇을 볼 수 있는가

쓰기:

어디까지 바꿀 수 있는가

외부 행동:

무엇은 승인받아야 하는가

중단·복원 경로까지 정합니다

조용하다고
잘 돌아가는 것은 아닙니다

01

중단 / 대기 / 재시도를 구분

02

성공뿐 아니라 실패도 알림

03

체크포인트에서 다시 시작

04

반복 실행이 중복 결과를 만들지 않게

합성 응답은
탐색 도구이지 여론의 대리인이 아닙니다

유용: 질문 점검 / 가설 탐색 / 시나리오 비교

주의: 모델 편향 / 조건 민감성 / 응답 의존성

실제 사람 자료와의 검증은 별도입니다

좋은 기준은
검사 가능한 문장으로 씁니다

01

필수 항목이 빠지지 않았는가

02

근거가 원문에 있는가

03

불확실성을 따로 표시했는가

04

사람이 승인할 부분을 구분했는가

연구 업무는
관찰과 해석을 분리해서 맡깁니다

01 ‘승인된 자료에서 조건·단위·결과를 추출하라’

02 ‘원문 위치를 붙이고 누락을 표시하라’

03 ‘관찰과 해석을 분리하라’

04 ‘근거가 없는 결론은 생성하지 마라’

행정 업무는
규정의 버전과 승인자를 붙여 맡깁니다

01 ‘제공된 규정의 적용일과 조항을 확인하라’

02 ‘요청 내용을 체크리스트와 대조하라’

03 ‘예외와 추가 확인 사항을 분리하라’

04 ‘초안만 작성하고 발송·승인은 하지 마라’

부서 지식관리는 작은 업무에서 검증하고 넓힙니다

첫 달: 반복 질문과 자료 소유자 찾기

둘째 달: 한 업무의 맥락·기준·실행 묶기

셋째 달: 오류·재사용·인수인계로 평가

기관 사정에 맞춰 조정하는 90일 제안

AKM Index는
다섯 가지 운영 능력을 묻습니다

Prompt:	명확한 작업 요청
Context:	필요한 맥락의 관리
Harness:	도구·규칙·권한의 결합
Loop:	검증·반복·복구
Interop & Governance:	상호운용과 책임

모델을 고를 때는 순위보다 업무 조건을 봅니다

데이터를 넣어도 되는가

필요한 도구와 형식을 지원하는가

실제 과제에서 품질이 나오는가

비용·지연·관리 조건이 맞는가

Obsidian을 쓰지 않아도 이 원리는 가져갈 수 있습니다

01

기관이 승인한 저장소

02

다시 읽을 수 있는 문서

03

근거와 책임이 남는 절차

04

도구는 달라도 구조는 설계할 수 있습니다

‘보안 때문에 못 씁니다’에는 업무를 나누어 답합니다

공개자료와 비공개자료 분리

초안 작성과 외부 행동 분리

자동 확인과 최종 승인 분리

허용된 범위에서 작은 작업부터

직접 살펴볼 수 있는 공개 자산입니다

시스템 파일 | https://system.cmdspace.work

CMDS | https://github.com/johnfkoo951/cmds-vault

LLM Wiki | https://github.com/johnfkoo951/cmds-llm-wiki

AKM Index | https://akm.cmdspace.work

오늘의 이야기를
계속 따라오실 수 있습니다

CmdMD | https://cmdmd.cmdspace.work

9Yohan | https://9yohan.cmdspace.work

더배러 | https://thebetter.stibee.com/

월간옵시디언 | https://pbl.to/MrSU8P

작업을 맡기기 전
이 여섯 문장을 채워보세요

01 무엇을 끝내야 하는가 / 무엇을 읽어야 하는가

02 어떤 형식으로 남기는가 / 무엇이면 통과인가

03 어디까지 행동해도 되는가 / 언제 멈추는가

성공한 답뿐 아니라
실패한 조건도 자산으로 남깁니다

01

무엇이 일어났는가

02

왜 일어났는가

03

어떻게 복구했는가

04

다음에는 무엇을 검사할 것인가

이 강의의 근거와 더 읽을 자료

본인 프레임워크와 공개 자료: system.cmdspace.work

AKM 루브릭: akm.cmdspace.work

기관 역할: www.kbsi.re.kr

합성 페르소나 사례: 운영 방법만 소개