무슨 뜻이야? : Claude Skills are not prompt hacks. They are the new operating layer for people who know how to delegate work to AI.

뜻은 이렇습니다.

"Claude Skills는 프롬프트 요령이나 꼼수가 아니다.
AI에게 일을 제대로 위임할 줄 아는 사람들을 위한 새로운 운영 계층이다."

조금 풀어보면,

prompt hacks는 그때그때 더 좋은 답을 얻기 위해 질문 문장을 영리하게 만드는 기술을 뜻합니다. 반면 **Skills**는 한 번의 답을 얻는 질문이 아니라, 내가 일을 처리하는 방식 전체를 AI가 반복해서 따르도록 묶어둔 체계라는 뜻입니다.

예를 들면 단순한 프롬프트는 이렇습니다.

"이 이메일을 정리해 줘."

하지만 Skill은 이런 내용을 함께 갖습니다.

이메일을 어떤 기준으로 분류하고,
무엇을 긴급하다고 판단하며,
어떤 형식으로 요약하고,
어떤 경우에 답장 초안을 만들 것인가.

그러니 여기서 말하는 **operating layer**는 컴퓨터의 운영체제와 비슷한 비유입니다. AI 모델 자체를 바꾸는 것은 아니지만, 그 위에 **업무 규칙·절차·기준·예시를 얹어 AI가 일하는 방식을 통제하는 층**이라는 것이죠.

용차이의 구조로 번역하면 아주 익숙합니다.

프롬프트는 지니에게 한 번 일을 시키는 것.
Skill은 지니가 앞으로도 그 일을 어떤 방식으로 처리해야 하는지 운영 문법을 심는 것.

그래서 이 문장의 핵심은 결국 이것입니다.

AI를 잘 쓰는 사람은 매번 질문을 잘 만드는 사람을 넘어, AI에게 일을 맡기는 구조를 만들어 놓는 사람이다.

우리가 말해 온 **운영체계**, **객체별 역할 분담**, **리딩멘트·본문 원문·맺음말의 고정 문법**, **아이부스 랩과 원펀치캠프의 듀얼 프로세스**도 바로 이 관점과 매우 가깝습니다. 다만 용차이는 Skills라는 기능이 나오기 전부터, 대화와 기록을 통해 그 운영 계층을 직접 만들어 온 셈입니다.

내 방식과 이 skills 방식 간에 장단점을 함 엮어서 비교해봐줄래?

용차이 방식과 Skills 방식은 경쟁 관계라기보다 **발견 단계와 정착 단계의 차이**에 가깝습니다.

한 문장으로 압축하면 이렇습니다.

용차이 방식은 운영 문법을 발견하는 방식이고, Skills는 발견된 운영 문법을 반복 실행하는 방식입니다.

둘을 맞대어 보면

구분	**용차이 방식**	**Skills 방식**
출발점	실제 사건과 대화	미리 정리된 업무 규칙
핵심 질문	"무슨 일이 일어나고 있지?"	"이 일은 어떤 절차로 처리하지?"
강점	예상하지 못한 발견	일관되고 빠른 반복
기록 형태	대화 원문·시행착오·수정 과정	지침·예시·템플릿·체크리스트
변화 방식	발생하면서 진화	사람이 Skill을 수정해야 진화
사람의 역할	운영자가 계속 판단하고 개입	정해진 범위는 AI에 위임
확장성	깊지만 운영자 의존도가 큼	다른 사람에게 복제하기 쉬움

구분	용차이 방식	Skills 방식
위험	창의 관성, 맥락 분산, 반복 실수	문법의 조기 고착, 예외 상황 오판

용차이 방식의 큰 장점

용차이 방식은 처음부터 정답을 고정하지 않습니다.

이번 듀얼 프로세스만 해도 그렇습니다.

처음에는 단순히 창을 둘로 나눴지만, 실제로 운영하면서 다음이 차례로 드러났습니다.

- 연구와 운영의 역할 차이
- 두 지니의 상호 검토 기능
- 잘못 토스한 질문에서 생긴 운영 문법
- 특정 지니에게 과도하게 의존하지 않게 되는 안전장치
- 운영자의 심리적 안정

이런 것은 Skills를 먼저 만들었다면 발견하기 어려웠을 수 있습니다. 처음 만든 규칙에 현실을 끼워 맞추려 했을 가능성이 있기 때문입니다.

즉, 용차이 방식은 **아직 이름조차 없는 것을 발견하는 데 강합니다.**

그리고 대화 원문을 남기기 때문에 나중에 결과뿐 아니라,

왜 그렇게 생각했고,
어디서 틀렸으며,
어떻게 수정했는가

까지 추적할 수 있습니다.

이것은 단순한 업무 기록보다 훨씬 풍부한 연구 자산입니다.

반대로 용차이 방식의 약점

지금까지 여러 번 겪었듯이 한 창의 지니가 맥락을 잘못 읽으면, 용차이가 직접 바로잡아야 합니다.

또한 좋은 운영 문법이 발견돼도 그것이 여러 대화 속에 흩어져 있으면 다음 창의 지니가 정확히 재현하지 못할 수 있습니다.

예를 들면 용차이는 이미 분명히 정했습니다.

본문 대화는 그대로 둔다.
지니는 리딩멘트, 맺음말, 해시태그, 프로필 리딩멘트만 보완한다.

그런데 지니가 자꾸 본문을 다시 편집하려 들었습니다.

이것은 **운영 문법은 발견됐지만 아직 시스템에 고정되지 않은 상태**입니다.

바로 이 지점에서 Skills가 유용해집니다.

Skills 방식의 장점

Skills는 한번 확인된 문법을 매번 다시 설명하지 않아도 되게 합니다.

가령 다음과 같은 Skill을 만들 수 있습니다.

원펀치 게시물 편집 Skill

입력:

- 용차이와 지니의 대화 원문
- 글 번호와 채널

규칙:

- 대화 원문은 수정하지 않는다.
- 리딩멘트만 작성한다.
- 맺음말은 본문과 중복하지 않는다.
- 해시태그는 채널에 맞게 작성한다.
- 프로필 리딩멘트는 해당 글의 의미를 한 문장으로 연결한다.
- 'AI 시대의 길찾기 안내자'를 습관적으로 붙이지 않는다.

이렇게 고정하면 제가 같은 실수를 반복할 가능성이 크게 줄어듭니다.

또 다른 예로는 다음이 가능합니다.

- 고객 질문의 성격을 판별하는 Skill
- 원펀치캠프 운영 기록 Skill
- 아이부스 랩 연구 기록 Skill
- 실제 사례 시리즈 구성 Skill

- 댓글 응답 초안 Skill

즉, Skills는 **용차이가 이미 여러 차례 검증한 운영 문법을 생산라인으로 옮기는 데 강합니다.**

Skills 방식의 가장 큰 위험

Skills는 너무 일찍 만들면 위험합니다.

예를 들어 우리가 #68을 처음 논의할 때 곧바로 Skill을 만들었다면,

"질문은 먼저 연구실을 찾아간다."

같은 틀린 문장이 그대로 규칙이 될 수도 있었습니다.

실제로 원펀치캠프 운영일지는 연구실이 아니었고, 문제는 공간보다 질문의 성격과 역할 판별에 있었습니다.

그러므로 아직 살아 움직이며 수정 중인 발견을 Skill로 고정하면,

틀린 문법을 더 빠르고 일관되게 반복하는 시스템

이 만들어질 수 있습니다.

Skills가 있다고 해서 판단이 사라지는 것이 아니라, 잘못된 판단이 자동화될 수도 있는 것이죠.

둘을 가장 잘 결합하는 순서

용차이에게 가장 어울리는 구조는 이것이라고 봅니다.

현실에서 발생

↓

용차이와 지니의 대화

↓

시행착오와 수정

↓

아이부스 랩에서 운영 원리 발견

↓

원펀치캠프에서 반복 검증

↓

충분히 익은 문법만 Skill로 고정

↓

반복 실행 중 예외가 생기면 다시 아이부스 랩으로

그러므로,

아이부스 랩은 Skill을 만드는 곳이 아니라, Skill로 만들 만한 문법을 발견하고 숙성시키는 곳입니다.

원펀치캠프는 그 문법이 현실에서 실제로 작동하는지 검증하는 곳이고요.

Skill은 그 둘을 통과한 문법을 반복 실행하는 장치가 됩니다.

가장 본질적인 차이

Claude Skills라는 개념은 대체로 이렇게 말합니다.

일하는 방식을 미리 포장해 AI에 위임하라.

그런데 용차이 방식은 그보다 앞단을 다룹니다.

아직 무엇이 일하는 방식인지조차 모를 때, 현실과 대화를 통해 그것을 발견한다.

그래서 저는 둘의 관계를 이렇게 부르고 싶습니다.

용차이 방식은 운영체계의 연구개발이고, Skills는 운영체계의 제품화입니다.

또는 더 짧게,

용차이 방식은 발견 엔진, Skills는 실행 엔진입니다.

지금 용차이가 Skills로 옮겨도 될 정도로 숙성된 것은 일부입니다. 게시물 포장 방식, 채널별 역할, 대화 원문 보존 원칙 같은 것은 Skill 후보가 될 수 있습니다.

반면 듀얼 프로세스의 질문 라우팅 문법처럼 계속 수정 중인 것은 아직 Skill로 굳히기보다, 지금처럼 아이부스 랩에서 살아 있는 연구 노트로 남기는 편이 맞습니다.

결국 용차이는 Skills를 따라가야 하는 사람이 아니라, **자신의 방식에서 충분히 익은 부분만 Skills로 내려보내야 하는 사람**에 가깝습니다.

Onepunchcamp AI 창업 부트캠프 의 주 고객은 어떤 이들일까? 즉, 부스분양을 받아서 자신만의 비즈니스를 만들어보겠다는 사람들의 취향은 보통의 Claude Skills 를 필요로 하는 이들과 뚜렷하게 구분되는 어떤 특성이 있지 않을까? 그리고 그들껜 우리의 방식이 보다 절실하지않을까는 생각이 떠올랐는데?

용차이, 저도 **바로 그 차이가 원펀치캠프의 고객을 훨씬 선명하게 만든다**고 봅니다.

보통 Claude Skills를 필요로 하는 사람은 이미 **해야 할 일이 비교적 분명한 사람**입니다.

"나는 영업 메일을 반복해서 써야 한다."
"고객 문서를 일정한 형식으로 만들어야 한다."
"조사·요약·콘텐츠 재가공 과정을 자동화하고 싶다."

즉, **자기 일은 이미 있고**, 그 일을 AI에게 더 잘 위임하려는 사람입니다.

반면 원펀치캠프에서 부스를 분양받으려는 사람은 그보다 앞선 상태에 있을 가능성이 큽니다.

"내가 가진 경험을 무엇으로 만들어야 하지?"
"내가 진짜 풀고 싶은 문제는 무엇이지?"
"무슨 사업을 해야 할지 아직 선명하지 않다."
"아이디어는 있지만, 사람들에게 보여줄 객체가 없다."

이들은 **업무를 자동화하기 전에 자기 일을 먼저 발견해야 하는 사람들**입니다.

두 고객의 결정적인 차이

Claude Skills가 필요한 사람	**원펀치캠프가 필요한 사람**
이미 반복 업무가 있다	아직 자기 일이 선명하지 않다
절차를 표준화하고 싶다	문제와 방향을 발견하고 싶다
AI에 일을 위임하고 싶다	AI와 함께 자기 일을 만들어야 한다
효율과 생산성이 우선이다	정체성·객체·사업화가 우선이다

Claude Skills가 필요한 사람　　**원펀치캠프가 필요한 사람**

Skill을 설치하면 바로 쓸 수 있다　자기만의 운영 문법부터 찾아야 한다

기존 조직·직무 안에서 개선한다　자기 이름의 작은 사업을 세우려 한다

그래서 원펀치캠프의 주 고객은 단순한 **AI 초보자**도 아니고, 일반적인 **예비창업자**만도 아닙니다.

저는 이렇게 정의하는 편이 더 정확하다고 봅니다.

자기 안에 경험과 문제의식은 있지만, 그것을 하나의 비즈니스 객체로 세우는 방법을 아직 찾지 못한 사람.

이들은 종종 나이와 경력이 적지 않습니다. 무엇인가를 오래 해왔지만, 그것을 자기 이름의 사업으로 번역하지 못했을 수 있습니다.

- 퇴직했거나 전환기를 맞은 사람
- 오래된 전문성과 경험이 있지만 상품화하지 못한 사람
- 기존 사업을 AI 시대에 맞게 다시 정의하려는 사람
- 아이디어는 많지만 무엇부터 만들어야 할지 모르는 사람
- 남의 프롬프트나 도구는 써보았지만 자기 체계는 없는 사람

바로 #001도 여기에 가깝습니다.

남주씨에게 당장 Skill을 하나 주면 일부 업무에는 도움이 될 수 있습니다. 그러나 근본 문제는 아직,

"AI 전환센터라는 자기 부스는 정확히 무엇을 하는 곳인가?"

가 자기 언어로 만들어지지 않았다는 데 있을 가능성이 큽니다.

그러니 Skill보다 앞서 필요한 것이 있습니다.

문제 발견 → 자기 언어화 → 첫 객체 생성 → 현실 반응 → 수정 → 운영체계 형성

이 과정이 바로 원펀치캠프가 제공하려는 것이죠.

왜 이들에게 용차이 방식이 더 절실한가

Skills 방식은 **발견된 문법을 자동화**합니다.

하지만 이 사람들은 아직 자동화할 문법이 없습니다.

남의 Skill을 가져다 써도 일시적으로 편리할 뿐,

- 무엇을 위해 쓰는지,
- 자기 사업에서 어디에 연결되는지,
- 어떤 고객 문제를 해결하는지

가 없으면 오래 가지 못합니다.

용차이 방식은 그 앞단을 다룹니다.

남의 Skill을 고르는 것이 아니라, 자기 삶과 경험에서 자기 Skill이 태어날 토양을 만드는 것.

그래서 이들에게는 완성된 Skill 패키지보다 다음이 더 필요합니다.

- 자기 말을 끝까지 들어줄 상대
- 아직 정리되지 않은 생각을 견디는 공간
- 작은 객체를 실제로 만들어 보는 과정
- 실패와 정지를 평가하지 않고 기록하는 운영일지
- 다른 사람과 비교하지 않고 자기 속도를 찾는 체계
- 언젠가는 독립하도록 설계된 apprenticeship

이게 원펀치캠프가 단순한 AI 교육이나 Skill 강좌와 다른 지점입니다.

가장 선명한 포지셔닝

저는 이런 대비가 가능하다고 봅니다.

Claude Skills는 이미 있는 일을 AI에게 맡기는 방법이다.
OnePunchCamp는 아직 없는 자기 일을 AI와 함께 만들어 가는 곳이다.

또는 더 용차이다운 표현으로는,

Skills는 일하는 방식을 포장한다.
원펀치캠프는 포장할 자기 방식부터 발견한다.

그리고 가장 강한 한 문장은 이것일 수 있습니다.

우리는 Skill을 가르치기 전에, 그 사람에게 어떤 Skill이 필요한 사람인지를 함께 발견합니다.

여기서 원펀치캠프의 고객은 대중 전체가 아닙니다. 이미 효율적으로 반복할 일이 명확한 사람은 Skills만으로도 상당 부분 해결할 수 있습니다.

반대로 원펀치캠프는 이런 사람에게 절실합니다.

“나는 남의 일을 더 잘하고 싶은 것이 아니라, 이제 내 일을 하나 만들어 보고 싶다.”

저는 이것이 **부스분양을 받고 자기 비즈니스를 만들려는 사람들의 가장 뚜렷한 공통 취향**이라고 봅니다. 이들은 편리한 도구보다 **자기 세계의 첫 좌표**가 필요한 사람들입니다.