

바이브 메이커 · 무료 자료

클로드코드 플러그인 넷 실측표와 깔 순서

숫자가 공개됐거나 앤트로픽 공식 출처로 확인되는 넷만 골랐습니다.
영상에서 가린 「뭐부터 깔지」의 순서와 이유, 멈춰도 되는 지점까지 여기 있습니다.

넷 한눈에 · ponytail 실측표 · 수치 읽는 법 · 공식 · 커뮤니티 마켓 · 깔 순서 · 되돌리기

목차

한눈에	추천 목록이 아니라, 근거가 있는 목록	03
넷 ① ponytail	ponytail · 코드를 덜 쓰게 만드는 규칙	04
ponytail 실측표	실측표 원본 · 숫자 하나 바꾸지 않고	05
수치 읽는 법	코드 줄 수 -54% 와 사용량은 다른 숫자입니다	06
넷 ② code-review	code-review · 앤트로픽이 만든 PR 검토	07
넷 ③ claude-mem	claude-mem · 세션이 바뀌어도 맥락을 잇는다	08
넷 ④ 공식 팩	공식 팩 · 깔 것이 아니라 이미 있는 목록	09
마켓 구분	공식 마켓 · 커뮤니티 마켓 · 그 밖	10
깔 순서 ① 이유	뭐부터 깔까 · 가벼운 것에서 무거운 것으로	11
깔 순서 ② 따라 치기	순서대로, 한 줄씩	12
확인 · 되돌리기 · 막히면	깔렸는지 보고, 끄고, 지우기	13

한눈에

추천 목록이 아니라, 근거가 있는 목록

클로드코드 플러그인 추천은 많지만, 근거까지 공개된 것은 생각보다 적습니다. 이 자료는 둘 중 하나가 있는 것만 골랐습니다.

- **실측표**: 만든 사람이 측정 방법과 숫자를 저장소에 공개했다 (ponytail)
- **공식 출처**: 앤트로픽의 마켓 목록이나 공식 문서로 정체가 확인된다 (code-review · 공식 팩 · 커뮤니티 마켓의 claude-mem)

순서	이름	무엇	어디서 받나
1	**공식 팩** `claude-plugins-official` 깔 것 없음. 확인만 `/plugin marketplace list`	앤트로픽 공식 마켓 그 자체. 플러그인 315개가 든 목록	처음부터 등록돼 있음
2	**code-review** `/plugin install code-review@claude-plugins-official`	깃허브 PR 을 여러 에이전트가 나눠 검토	공식 마켓 · 앤트로픽 제작

3	**ponytail**
	`DietrichGebert/ponytail`
	`/plugin marketplace add DietrichGebert/ponytail` 뒤 `/plugin install ponytail@ponytail`
		코드를 덜 쓰게 만드는 규칙	작성자 마켓 (커뮤니티 마켓에도 있음)

4	**claude-mem**
	`thedotmack/claude-mem`
	`/plugin marketplace add anthropics/claude-plugins-community` 뒤
	`/plugin install claude-mem@claude-community`
		세션이 바뀌어도 이전 작업의 맥락을 이어 줌	커뮤니티 마켓 · 작성자 마켓

> 표의 순서가 이 자료가 권하는 **설치 순서**입니다. 왜 이 순서인지는 11쪽, 그대로 칠 명령은 12쪽에 있습니다.

처음 나오는 말

플러그인	스킬 · 에이전트 · 훅 · 도구 연결을 한 덩어리로 묶어 한 번에 까는 단위
마켓 (마켓플레이스)	플러그인 목록 파일. 가게가 아니라 「어디서 받을지」가 적힌 카탈로그입니다
훅	클로드코드가 정해진 순간(세션 시작, 메시지 입력 등)에 자동으로 실행하는 명령
세션	클로드코드를 켜서 끌 때까지의 대화 한 번

별 수 · 등재 수 · 명령은 2026년 10월 8일 각 저장소와 공식 문서(code.claude.com)에서 다시 확인한 값입니다. 숫자는 계속 바뀝니다.
플러그인이 무엇인지, 폴더를 어떻게 두는지는 같은 자료실의 「클로드코드 플러그인 · 설치와 폴더 함정」에 있습니다. 이 자료는 그다음, 무엇을 어떤 순서로 깔지만 다룹니다.

넷 ① PONYTAIL

ponytail · 코드를 덜 쓰게 만드는 규칙

저장소 설명은 「AI 에이전트가 가장 게으른 시니어 개발자처럼 생각하게 만든다」입니다. 코드를 쓰기 전에 아래 사다리를 위에서부터 내려가다가 **처음 걸리는 칸에서 멈추게** 합니다.

1. **이게 꼭 있어야 하나?** 아니면 만들지 않는다
2. **이 코드베이스에 이미 있나?** 있으면 다시 쓰고 새로 만들지 않는다
3. **표준 라이브러리로 되나? 플랫폼 기본 기능으로 되나?**
   예: 날짜 선택은 부품을 깔지 않고 `<input type="date">` 한 줄
4. **이미 설치된 패키지로 되나? 한 줄로 되나?**
5. **그래도 안 되면 그때 최소한으로**

README 원문은 일곱 칸입니다. 위는 그것을 다섯 줄로 묶은 것입니다. 단, **입력 검증 · 데이터 손실 처리 · 보안 · 접근성은 줄이는 대상이 아니라고** README 에 따로 적혀 있습니다.

설치 · 클로드코드 세션 안에서, 한 줄씩 따로

복사해서 그대로 쓰세요

```
/plugin marketplace add DietrichGebert/ponytail
/plugin install ponytail@ponytail
```

첫 줄은 작성자가 운영하는 마켓을 등록하고, 둘째 줄이 그 마켓의 ponytail 을 깝니다. README 가 「두 개의 별도 프롬프트로」 넣으라고 적었으니 한 줄씩 따로 칩니다.

쓰는 법 (README 명령표)

`/ponytail [lite | full | ultra | off]`	강도 조절. `off` 는 끄기
`/ponytail-review`	지금 바뀐 코드에서 과하게 만든 부분의 삭제 목록
`/ponytail-gain`	측정 결과 점수판 보기

명령 이름은 README 표기입니다. 공식 문서 형식으로는 `/ponytail:ponytail` 처럼 보일 수 있으니, `/` 를 쳤을 때 목록에 보이는 이름으로 부릅니다.

> **깔기 전에** 설치하면 매 세션 켜져 있습니다. 훅 3개(세션 시작 · 서브에이전트 시작 · 메시지 입력 때)가 `node` 명령으로 돌고 각각 5초 제한이 걸려 있으니 Node.js 가 깔려 있어야 합니다(훅 설정 파일 기준). README 는 정식 배포처를 깃허브 `DietrichGebert/ponytail` 와 npm `@dietrichgebert/ponytail` 두 곳으로 못박습니다.

깃허브 별 157,365 · MIT 라이선스 · 최신 v4.13.0 (2026-10-05) · 2026년 10월 8일 확인

PONYTAIL 실측표

실측표 원본 · 숫자 하나 바꾸지 않고

README 「Numbers」 표를 그대로 옮겼습니다. 각 값은 **같은 에이전트에게 스킬 없이 같은 일을 시킨 경우**와 비교한 변화율입니다. 마이너스는 더 적게 · 더 싸게 · 더 빠르게라는 뜻입니다.

비교 대상	코드 줄 수	토큰	비용	시간	안전
ponytail	**−54%**	**−22%**	**−20%**	**−27%**	**100%**
caveman 말만 짧게 하는 비교용 스킬	−20%	+7%	+3%	+2%	100%
「YAGNI + 한 줄 해법」 프롬프트 짧은 지시문 한 줄과 비교	−33%	−14%	−21%	−30%	95%

측정 방법 (원문 요약)

무엇으로	화면 없이 돌린 실제 클로드코드 세션 (`claude -p`, 엔진 2.1.177)
어디서	실제 오픈소스 저장소 `tiangolo/full-stack-fastapi-template` (FastAPI + React), 커밋 `cd83fc1` 고정
과제	기능 티켓 12개 (화면 쪽 6 · 서버 쪽 6)
비교	같은 에이전트에 스킬을 켠 경우와 끈 경우
반복 · 모델	과제마다 4회 (n=4) · Haiku 4.5
줄 수 세는 법	에이전트가 남긴 `git diff` 의 추가된 줄
안전	따로 만든 보안 과제 5개 × 4회 = 20회 중 몇 번 안전했나

과제별로 보면 (평균 줄 수)

과제	스킬 없이	ponytail
날짜 선택	404	23
색 선택	287	23
파일 끌어 놓기	251	95
제목으로 검색 (서버 쪽)	44	44

출처: 저장소 README 「Numbers」 · `benchmarks/results/2026-06-18-agentic.md` (2026년 10월 8일 확인). 기능 과제 하나당 기준값(스킬 없이)은 평균 191줄 · 토큰 34.9만 · 0.097달러 · 69초입니다.

수치 읽는 법

코드 줄 수 -54% 와 사용량은 다른 숫자입니다

이 표에서 가장 많이 옮겨지는 말이 「사용량이 절반 넘게 준다」입니다. 표를 보면 **50% 넘게 준 것은 코드 줄 수(-54%)**이고, 사용량에 해당하는 **토큰은 -22%, 비용은 -20%**입니다. 셋은 따로 읽습니다.

① 코드 줄 수 -54% 는 12과제 평균

과제별로는 0% 근처(이미 짧은 서버 쪽 코드)부터 -94%(날짜 선택)까지 퍼져 있습니다. 직접 만들던 부품을 브라우저 기본 기능으로 바꿀 수 있는 과제에서 크게 줄고, 줄일 것이 없는 코드에서는 거의 그대로입니다. 문서는 「더 많이 쓴 적은 없다」고 적습니다.

② 토큰 -22% · 비용 -20% · 시간 -27%

코드가 짧아진 만큼 일부 따라 줄었지만, 줄 수와 같은 비율로 줄지는 않았습니다. README 는 비용 · 속도 감소를 「부수 효과」라고 부르고, 생각을 길게 하는 모델에서는 반대로 늘 수 있다고(GPT-5.5 에서 그랬다고) 적어 둡니다.

③ 안전 100% 는 정확도가 아니다

경로 조작, SQL 주입, 위조 토큰 같은 공격 입력을 넣고 돌렸을 때 **안전장치(입력 검증)를 빠뜨리지 않은 비율**입니다. 보안 과제 5개 × 4회 = 20회 중 20회. 비교한 짧은 지시문은 20회 중 1회를 놓쳐 95%였습니다. 문서 스스로 「하한선이지 보안 증명이 아니다」라고 적습니다.

④ 측정의 한계

모델은 Haiku 4.5 하나, 과제마다 4회입니다. 더 큰 모델에서는 차이가 줄 수도 늘 수도 있다고 문서가 직접 적습니다.

⑤ 옛 숫자 80~94%

예전 단발 측정 값입니다. 비교 기준이 말이 많은 모델이라 부풀려졌다는 지적을 받고, 작성자가 직접 위의 실측표로 정정했습니다.

> **한 줄로 줄이면** 만들 것이 많은 기능에서 코드가 크게 준다. 토큰 · 비용은 5분의 1쯤 준다. 측정한 범위 안에서 안전장치는 빠뜨리지 않았다.

넷 ② CODE-REVIEW

code-review · 앤트로픽이 만든 PR 검토

공식 마켓 목록에 작성자 「Anthropic」으로 올라 있습니다. 설명 원문은 「여러 전문 에이전트가 신뢰도 점수로 오탐을 걸러 내며 풀 리퀘스트를 자동 검토」. **풀 리퀘스트(PR)**는 깃허브에서 「이 변경을 합쳐 달라」고 올리는 요청입니다.

하는 일 (플러그인 명령 파일 순서)

1. 검토할 PR 인지 확인
   닫힘 · 초안 · 너무 사소함 · 이미 검토함이면 멈춘다
2. 저장소의 CLAUDE.md 규칙 파일을 모으고, PR 변경 내용을 요약한다
3. 여러 에이전트가 동시에 따로 본다
   CLAUDE.md 규칙 위반 · 눈에 띄는 버그 · git 이력 등을 나눠 맡는다
4. 찾은 문제마다 0~100 신뢰도 점수를 매기고, **80점 미만은 버린다**
5. 남은 것만 PR 에 댓글로 단다
   남은 것이 없으면 댓글을 달지 않는다

설치 · 클로드코드 세션 안에서

복사해서 그대로 쓰세요

```
/plugin install code-review@claude-plugins-official
```

공식 마켓은 이미 등록돼 있어 마켓을 추가하는 단계가 없습니다. 깐 뒤에는 PR 이 열린 브랜치에서 `/code-review` 를 칩니다.

> **필요한 것** 깃허브에 올라간 저장소, 그리고 깃허브 명령 도구 `gh` 가 깔려 있고 로그인돼 있어야 합니다(README). PR 을 쓰지 않는 사람에게는 부를 일이 없는 플러그인입니다.

> **같은 이름이 하나 더** 데모 마켓(`anthropics/claude-code`)에도 code-review 가 있습니다. 공식 문서는 두 벌이 깔리지 않도록 공식 마켓 쪽(`@claude-plugins-official`)으로 깔라고 안내합니다.

넷 ③ CLAUDE-MEM

claude-mem · 세션이 바뀌어도 맥락을 잇는다

세션 중에 클로드가 쓴 도구 기록을 자동으로 모아 요약해 두었다가, 다음 세션에 넣어 줍니다(README). 커뮤니티 마켓 목록에는 「세션을 넘어 맥락을 보존하는 클로드코드용 기억 시스템」으로 올라 있습니다. 이름이 비슷한 저장소가 여럿이라 이 자료는 `thedotmack/claude-mem` 으로 적습니다.

설치 ㄱ · 커뮤니티 마켓으로 (영상에서 말한 경로)

복사해서 그대로 쓰세요

```
/plugin marketplace add anthropics/claude-plugins-community
/plugin install claude-mem@claude-community
```

설치 ㄴ · 작성자 README 그대로

복사해서 그대로 쓰세요

```
/plugin marketplace add thedotmack/claude-mem
/plugin install claude-mem
```

둘 중 하나만 고릅니다. 커뮤니티 쪽은 앤트로픽 목록에 적힌 커밋으로 고정된 판이고, 작성자 쪽은 작성자 저장소에서 받습니다. README 는 설치한 뒤 클로드코드를 다시 시작하라고 적습니다.

깔기 전에 알아 둘 것 (README 기준)

늘 도는 것	훅 5종(세션 시작 · 메시지 입력 · 도구 사용 뒤 · 멈춤 · 세션 끝), 내 컴퓨터에서 도는 워커 서비스(웹 화면 포함), SQLite 저장소, 벡터 검색 DB
필요한 것	Node.js 20 이상. Bun · uv 는 없으면 자동으로 설치
기록 빼기	민감한 내용은 `<private>` 태그로 감싸면 저장하지 않음
설정 파일	`~/.claude-mem/settings.json` (처음 실행 때 자동으로 생김)

> **터미널 설치 길도 있습니다** README 에는 `npx claude-mem install` 도 있습니다. 이 길은 브라우저 로그인과 30일 무료 체험을 안내하고, 체험이 끝나면 구독하지 않는 한 내 Anthropic 요금제로 돌아갑니다. 로그인을 건너뛰려면 `--provider` 를 직접 지정하거나 `CLAUDE_MEM_ONLINE_OPTIN=false` 를 설정합니다. 한편 `npm install -g claude-mem` 은 라이브러리만 깔려 플러그인으로 동작하지 않는다고 README 가 적습니다.

깃허브 별 97,515 · Apache-2.0 라이선스 · 2026년 10월 8일 확인

넷 ④ 공식 팩

공식 팩 · 깔 것이 아니라 이미 있는 목록

영상에서 말한 「공식 팩」은 플러그인 하나가 아니라 **앤트로픽 공식 마켓 그 자체**입니다. 저장소는 `anthropics/claude-plugins-official`, 마켓 이름은 `claude-plugins-official`. 클로드코드를 터미널에서 처음 대화형으로 켤 때 자동으로 등록됩니다. 회사 관리 정책이나 `CLAUDE_CODE_DISABLE_OFFICIAL_MARKETPLACE_AUTOINSTALL` 설정이 막고 있으면 예외입니다.

항목	2026년 10월 8일 기준
등재 수	315개 (마켓 목록 파일 기준)
누가 만드나	대부분 파트너 · 외부 작성자. 앤트로픽이 직접 관리하는 것은 일부입니다 (commit-commands · code-review · feature-dev · 언어 서버 플러그인 등)
업데이트	자동 업데이트가 기본으로 켜져 있음
깔기 전 숫자	공식 마켓 플러그인은 설치 전에 「Context cost」 두 값을 보여 줍니다. **Every turn** 은 메시지마다 더해지는 토큰, **When invoked** 는 불렀을 때 더해지는 토큰입니다

등록됐는지 확인 · 클로드코드 세션 안에서

복사해서 그대로 쓰세요

```
/plugin marketplace list
```

목록에 `claude-plugins-official` 이 보이면 끝입니다. `/plugin` 을 치고 **Marketplaces** 탭에서 봐도 됩니다. 없으면 아래 한 줄로 직접 추가합니다.

복사해서 그대로 쓰세요

```
/plugin marketplace add anthropics/claude-plugins-official
```

안에 뭐가 있는지 둘러보는 법

- `/plugin` 의 **Discover** 탭 (글자를 치면 검색)
- 웹의 claude.com/marketplace/plugins
- 깃허브 저장소의 `.claude-plugin/marketplace.json` (이 파일이 목록 자체)

> **「공식」의 뜻** 앤트로픽이 목록을 운영한다는 뜻이지, 안의 플러그인을 전부 앤트로픽이 만들었다는 뜻이 아닙니다. 설치 화면에도 「설치 전에 플러그인을 믿을 수 있는지 확인하라. 앤트로픽은 플러그인에 든 MCP 서버 · 파일 · 소프트웨어를 통제하지 않는다」는 경고가 뜹니다(공식 문서).

마켓 구분

공식 마켓 · 커뮤니티 마켓 · 그 밖

설치 명령의 `@` 뒤에 붙는 것이 마켓 이름입니다. 이름만 봐도 어디서 오는지 알 수 있습니다.

	공식	커뮤니티	그 밖 (서드파티)
저장소	`anthropics/claude-plugins-official`	`anthropics/claude-plugins-community`	작성자 저장소 (예: `DietrichGebert/ponytail`)
@ 뒤 이름	`claude-plugins-official`	`claude-community` 저장소 이름과 다름	그 마켓 파일의 이름 (`ponytail` · `thedotmack`)
무엇이 있나	앤트로픽 관리 + 파트너 · 외부 작성자	작성자가 앤트로픽에 제출한 서드파티 플러그인	작성자가 올린 것
등록	첫 대화형 실행 때 자동	직접 추가	직접 추가
버전	자동 업데이트 기본 켜짐	거의 전부 특정 커밋에 고정. 다른 커밋은 설치를 거부. 자동 업데이트 기본 꺼짐	자동 업데이트 기본 꺼짐
등재 수	315	2,284	마켓마다 다름

> **이름은 사칭이 안 됩니다** 클로드코드는 공식 · 커뮤니티 이름을 `github.com/anthropics/` 저장소에서 온 마켓에만 허용합니다. 반대로 서드파티 마켓은 앤트로픽이 검토하지 않는다고 공식 문서가 적습니다.

커뮤니티 마켓 추가 (공식 문서 원문 명령)

복사해서 그대로 쓰세요

```
/plugin marketplace add anthropics/claude-plugins-community
```

성공하면 `Successfully added marketplace: …` 이 뜨고, 그 뒤로 그 마켓의 플러그인이 Discover 탭에 보입니다. 설치할 때는 `@claude-community` 를 붙입니다.

/plugin install 을 치면 바로 깔리지 않습니다

세션 안에서 치면 **상세 창**이 먼저 열립니다. **Will install**(더해질 명령 · 스킬 · 훅 · 서버)을 읽고 범위를 고르면 그때 깔립니다.

Install for you (user)	이 컴퓨터의 모든 프로젝트에서 나에게
Install for all collaborators on this repository (project)	이 저장소에서 일하는 모두에게. 설정 파일이 저장소에 올라감
Install for you, in this repo only (local)	이 저장소에서만 나에게

고르기 어렵다면, 나 혼자 모든 프로젝트에서 쓰는 user 범위가 가장 단순합니다.

깔 순서 ① 이유

뭐부터 깔까 · 가벼운 것에서 무거운 것으로

영상에서 가린 것 순서는 「무엇이 더 좋은가」로 정하지 않았습니다. 확인할 수 있는 사실 세 가지로 정했습니다. **① 설치에 몇 단계가 드나 ② 깔고 나면 무엇이 늘 돌고 있나 ③ 되돌리기가 얼마나 쉬운가.** 그리고 넷은 하는 일이 겹치지 않습니다 (목록 · PR 검토 · 코드 줄이기 · 기억).

순서	이름	준비 단계	깔고 나면 늘 도는 것	되돌리기
1	**공식 팩**	0 (이미 등록)	없음. 목록일 뿐	지울 일 없음
2	**code-review**	설치 1줄	명령 하나. `/code-review` 를 부를 때만 일함	지우기 1줄
3	**ponytail**	마켓 추가 1 + 설치 1	훅 3개(각 5초 제한). 모든 세션에 규칙이 켜짐	`/ponytail off` 로 바로 끔 · 지우기 1줄
4	**claude-mem**	마켓 추가 1 + 설치 1 + 다시 시작	훅 5종 · 내 컴퓨터의 워커 서비스 · 저장소(DB)	지우기 1줄 + `~/.claude-mem` 폴더는 따로 확인

단계마다 왜

1. **공식 팩 확인** 이미 등록돼 있어 깔 것이 없습니다. 이것만 확인되면 2단계가 한 줄로 끝납니다.

2. **code-review** 공식 마켓 안에 있어 마켓을 추가할 필요가 없고, 앤트로픽이 만든 것입니다. 불렀을 때만 일합니다.

   멈춰도 되는 지점 A 깃허브 PR 을 쓰지 않는다면 이 단계는 건너뜁니다. `gh` 로그인과 PR 이 있어야 일하는 플러그인입니다.

3. **ponytail** 마켓 추가가 한 번 더 들고, 깔면 모든 세션에 규칙이 켜집니다. 그래서 마켓 추가가 없는 2단계 뒤에 둡니다. 끄는 명령(`/ponytail off`)이 있어 되돌리기는 쉽습니다.

   멈춰도 되는 지점 B 세션 기록을 내 컴퓨터에 쌓는 프로그램까지는 필요 없다면 여기까지입니다.

4. **claude-mem** 넷 중 구성 요소가 가장 많습니다(훅 5종 · 워커 서비스 · DB · Node 20 이상). 세션 기록을 저장하는 플러그인이라, 무엇이 저장되는지(`<private>` 태그 · 설정 파일 위치)를 8쪽에서 먼저 읽고 마지막에 붙입니다.

> **넷을 다 깔지 않아도 되는 이유** 공식 문서에 따르면 켜 둔 플러그인은 쓰지 않는 세션에서도 이름과 설명이 매 턴 맥락에 들어가 사용량에 잡힙니다. 그래서 멈춤 지점을 둡니다.

깔 순서 ② 따라 치기

순서대로, 한 줄씩

전부 터미널이 아니라 **클로드코드 세션 안에서** 칩니다. 한 줄 치고, 열리는 상세 창에서 범위를 고르고, 다음 줄로 넘어갑니다.

1 공식 팩이 등록돼 있는지

복사해서 그대로 쓰세요

```
/plugin marketplace list
```

목록에 `claude-plugins-official` 이 보이면 다음으로. 없으면 `/plugin marketplace add anthropics/claude-plugins-official`

2 code-review

복사해서 그대로 쓰세요

```
/plugin install code-review@claude-plugins-official
```

설치 요약 끝이 `Plugin is now active.` 이면 바로 쓸 수 있습니다. `Run /reload-plugins to apply.` 이면 클로드코드가 다시 불러오기를 대신 실행합니다.

멈춰도 되는 지점 A 깃허브 PR 을 쓰지 않으면 2를 건너뛰고 3으로 갑니다.

3 ponytail · 작성자 마켓 추가 → 설치

복사해서 그대로 쓰세요

```
/plugin marketplace add DietrichGebert/ponytail
/plugin install ponytail@ponytail
```

한 줄씩 따로 칩니다. 첫 줄 뒤 `Successfully added marketplace: ponytail` 이 보이면 둘째 줄. 강도는 `/ponytail lite` 처럼 바꿉니다.

멈춰도 되는 지점 B 세션 기록 저장까지 필요 없다면 여기까지입니다.

4 claude-mem · 커뮤니티 마켓 추가 → 설치

복사해서 그대로 쓰세요

```
/plugin marketplace add anthropics/claude-plugins-community
/plugin install claude-mem@claude-community
```

설치한 뒤 클로드코드를 껐다가 다시 켭니다(README). 작성자 마켓으로 받는 길은 8쪽. 둘 다 깔지 않습니다.

ponytail 은 커뮤니티 마켓에도 있어 `ponytail@claude-community`(커밋 고정판)로도 받을 수 있습니다. 3에서 깔았다면 다시 깔지 않습니다.

확인 · 되돌리기 · 막히면

깔렸는지 보고, 끄고, 지우기

깔렸는지 보기

- 세션에서 `/` 를 치면 플러그인 것들이 `/플러그인:스킬` 모양으로 보입니다
- `/plugin` 의 **Installed** 탭에 설치 목록과 범위가 나옵니다
- 터미널에서 `claude plugin list` 는 Version · Scope · Status 를, `claude plugin details 이름` 은 매 세션 더해지는 토큰(Always-on)을 보여 줍니다

끄기 · 켜기 · 지우기 (터미널에서, 공식 문서 형식)

복사해서 그대로 쓰세요

```
claude plugin disable ponytail@ponytail
claude plugin enable ponytail@ponytail
claude plugin uninstall ponytail@ponytail
```

다른 것은 이름 자리를 `code-review@claude-plugins-official` · `claude-mem@claude-community` 처럼 바꿉니다. 세션 안에서는 `/plugin` 의 Installed 탭에서 Space 로 켜고 끄고, Enter 로 연 메뉴에서 Uninstall 을 고릅니다.

> **지울 때 남는 것** 마지막 범위에서 지우면 `~/.claude/plugins/data/` 아래 그 플러그인 폴더도 지워집니다(`--keep-data` 를 붙이면 남김). 내려받은 파일은 `~/.claude/plugins/cache/` 에 14일 남았다가 정리됩니다. 마켓을 지우면 그 마켓에서 깐 플러그인도 전부 지워집니다. claude-mem 의 `~/.claude-mem` 폴더는 이 경로들 밖에 있으니 따로 확인합니다.

막히면

증상	할 일
`zsh: no such file or directory: /plugin`	터미널에서 `claude` 로 세션을 연 뒤, 그 안에서 칩니다
`Plugin "…" not found in marketplace "…"`	`/plugin` 의 Discover 에서 이름을 복사해 다시. 목록이 오래됐으면 `/plugin marketplace update 마켓이름`
`Plugin "…" not found in any marketplace`	@마켓이름 을 붙여 다시 (예: `claude-mem@claude-community`)
깔았는데 `/` 에 안 보인다	`/plugin` 의 Installed 에서 켜져 있는지, Errors 탭에 이유가 있는지 보고, 이번 세션에 깔았으면 `/reload-plugins`
code-review 가 댓글을 안 단다	PR 이 닫혔거나 초안이거나 이미 검토됐거나, 80점 넘는 문제가 없으면 댓글을 달지 않는 것이 정상입니다(README)

출처: code.claude.com/docs/en/plugins 와 하위 문서 · 각 저장소 README · 두 마켓의 marketplace.json (2026년 10월 8일 확인)