

바이브 메이커 · 무료 자료

클로드 토큰 아끼는 무료 도구 둘

Headroom 과 Task Observer.
설치 방법과 링크를 한 곳에 모았습니다.

Headroom · 압축 | Task Observer · 기록 | 설치 명령 | 원본 링크

CONTENTS · 목차

7 PAGES

목차

한 장 요약	하나는 덜 보내고, 하나는 덜 반복하게 합니다	03
설치 1 · Headroom	설치하고, 클로드 코드를 wrap 으로 엽니다	04
설치 2 · Task Observer	깔고 끝이 아니라 켜는 블록까지 넣어야 돕니다	05
알고 쓰기	숫자는 작업마다 다르고, 반영은 내가 합니다	06
원본 링크	설치 전에 원본 README를 한 번 열어 보세요	07

한 장 요약

하나는 덜 보내고, 하나는 덜 반복하게 합니다

둘 다 무료 오픈소스입니다. Headroom 은 클로드에 넘어가는 내용을 줄이고, Task Observer 는 내가 고친 것을 기록해 **같은 지적을 덜 하게** 돕습니다.

Headroom · 압축

도구 결과 · 로그 · 파일을 클로드에 닿기 전에 압축합니다. **압축은 내 컴퓨터에서** 돌고, 내용을 어디로 보내지 않습니다.

저장소 소개: 코딩 에이전트 **토큰 20% 감소**, JSON **60~95% 감소**, 답은 그대로(same answers).

라이선스 **Apache 2.0**

Task Observer · 기록

작업 중 **내가 고친 것 · 드러낸 선호 · 스킬이 못 다루는 빈틈**을 기록합니다.

그걸로 **스킬 개선 제안**을 만듭니다. 스킬을 직접 고치지는 않고, **적용은 내가 검토**해서 합니다.

라이선스 **CC BY 4.0** · 클로드 스킬

📌 두 도구 모두 **다른 사람이 공개한 오픈소스**입니다. 바이브 메이커와 제휴 관계는 없습니다. 이 자료의 명령과 수치는 **2026-09-26 각 저장소 README 기준**입니다.

설치하고, 클로드 코드를 wrap 으로 엽니다

1 **설치 (파이썬 패키지)**
아래 둘 중 하나. **headroom 명령은 파이썬 패키지에만** 들어 있습니다

```
$ pip install "headroom-ai[all]"
또는
$ uv tool install --python 3.13 "headroom-ai[all]"
```

2 **클로드 코드를 Headroom 으로 감싸서 실행**
내 컴퓨터에 프록시를 켜고, 그 프록시를 거치게 설정된 클로드 코드가 열립니다. **세션을 열 때마다 이 명령으로** 엽니다

```
$ headroom wrap claude
```

3 **내 작업에서 얼마나 줄었는지 보기**
README 가 권하는 확인법입니다. 줄어드는 폭은 작업마다 다릅니다

```
$ headroom savings
```

⚠️ **wrap 이 하나를 더 설치합니다**

headroom wrap 은 코드 탐색 도구 **Serena** 를 설치해 사용자 범위(`~/.claude.json`)에 등록합니다. 그래서 다른 프로젝트에서도 남아 있습니다. 빼려면 `--code-memory none`, 되돌리려면 `headroom unwrap claude`.

💡 **npm 의 headroom-ai 는 코드에서 불러 쓰는 라이브러리**라 headroom 명령이 없습니다. 클로드 코드에 붙일 거면 위 파이썬 설치를 쓰세요.

깔고 끝이 아니라
켜는 블록까지 넣어야 돕니다

1 **스킬 설치 (둘 중 하나)**
명령 한 줄, 또는 저장소 ZIP 을 받아 **SKILL.md · references/ · scripts/** 를 함께 `.claude/skills/task-observer/` 에

```
$ npx skills add rebelytics/one-skill-to-rule-them-all --skill task-observer
```

2 **CLAUDE.md 에 활성화 블록 넣기**
README: 파일만 깔면 잘 안 불립니다. **references/environments.md** 의 「The activation block」을 붙여넣고 **[ABSOLUTE PATH]** 를 내 폴더 경로로 바꿉니다. 첫 문단 원문:

📋 **activation block · 첫 문단 (전문은 environments.md)** COPY → PASTE

```
Before the first tool call of any session — and before writing or proposing a plan, not merely before executing one — invoke the task-observer skill AND execute its Session Start Protocol (storage check, frontmatter scan, review trigger). Loading the skill and running the protocol are separate steps; a session that loads the file and stops has activated nothing. Any turn that will involve a tool call counts; do not classify the session as "too simple" from its opening message.
```

✅ **새 세션**에서 확인합니다. 작업을 몇 번 한 뒤 **skill-observations/observation-log/** 폴더가 생겼으면 켜진 겁니다. 설치한 그 세션에선 확인이 안 됩니다.

💡 **claude.ai 웹 · 앱**은 파일을 못 쓰니 세션 끝에 **정리 문서(handoff doc)**를 만들어 주는 방식으로 돕니다. 기록이 쌓이는 방식은 클로드 코드입니다.

숫자는 작업마다 다르고, 반영은 내가 합니다

Headroom · README 의 시나리오별 절감

시나리오	줄어든 토큰
코드 검색 결과 100개	21%
장애 대응 디버깅	57%
코드베이스 탐색	42%
깃허브 이슈 분류	30%

💡 README: **반복이 많은 데이터일수록** 많이 줄고(반복되는 JSON · 로그는 90% 넘게), **일반 문장이나 이미 촘촘한 출력은 거의 안 줄어듭니다.** 내 숫자는 `headroom savings` 로.

Task Observer · 알아둘 세 가지

제안만 해요 — 스킬을 직접 고치지 않습니다. **기록과 개선 제안**을 만들고, 반영은 내가 검토해서 합니다.

규모가 클수록 — README 의 정직한 경계: 스킬이 몇 개 안 되는 작은 셋업이면 **클로드 기본 메모리로도 비슷한 효과**를 볼 수 있습니다. 스킬이 늘수록 값어치가 커집니다.

직접 붙여넣기 — 클로드 코드가 이 블록을 사용자 설정에 **쓰길 거부**할 수 있다고 적혀 있습니다(남이 쓴 지시문이라서). 그럴 땐 **CLAUDE.md 에 직접 붙여넣으세요.**

설치 전에 원본 README를 한 번 열어 보세요

명령은 바뀔 수 있습니다. 이 자료와 다르면 **README 가 맞습니다.**

1 **Headroom · Apache 2.0**
`github.com/headroomlabs-ai/headroom`

2 **Task Observer · CC BY 4.0**
`github.com/rebelytics/one-skill-to-rule-them-all`
활성화 블록: `references/environments.md`

⚡ **순서 추천**

Headroom 먼저. 설치하고 wrap 으로 연 세션부터 압축이 걸립니다. Task Observer 는 **스킬을 쓰기 시작했을 때** 붙이면 기록이 쌓이는 만큼 효과가 납니다.

💬 따라 하다 막히면 **단톡방에 물어보세요.** 어디서 멈췄는지 화면만 올려주셔도 됩니다. 새 자료가 나오면 **@vibe.trender** 에 먼저 올립니다.