

바이브 메이커 · 무료 자료

클로드코드 플러그인 넷 설치 가이드

에이전트 스킬스 · 컨텍스트 세븐 · 트웬티퍼스트 · 스트릭스. 넷의 설치 명령 원문, 키 받는 자리, 무료로 되는 범위, 스트릭스 돌리기 전 준비 셋, 지우는 법까지 각 저장소 README 기준으로 정리했습니다.

설치 명령 원문 | 키 받는 자리 | 무료 범위 | 지우는 법

목차

한눈에	넷이 하는 일과 만드는 순서	03
준비물	키 받는 자리와 무료로 되는 범위	04
① 기획 · 설치	에이전트 스킬스, 클로드코드 안에서 두 줄	05
① 기획 · 쓰기	명령 아홉 개와 이름이 겹칠 때	06
② 문서	컨텍스트 세븐, 터미널에서 한 줄	07
③ 화면	트웬티퍼스트, 키를 받고 한 줄	08
④ 보안 · 준비	스트릭스 돌리기 전 준비 셋	09
④ 보안 · 실행	돌리기, 결과 보기, 클로드에 붙이기	10
순서대로 쓰기	넷을 이어 쓰는 요청 문장	11
지우기 · 막히면	지우는 법과 자주 막히는 곳	12

한눈에

넷이 하는 일과 만드는 순서

넷은 하는 일이 겹치지 않습니다. 기획 → 문서 → 화면 → 보안, 만드는 순서대로 하나씩 맡습니다.

순서	이름 · 저장소	하는 일	라이선스 · 별
기획	에이전트 스킬 `addyosmani/agent-skills`	기획부터 출시까지 단계에 맞는 스킬이 켜진다. 스킬 25개 · 명령 9개	MIT 약 11만 5천
문서	컨텍스트 세븐 `upstash/context7`	쓰는 라이브러리의 지금 버전 문서와 예제를 가져온다	MIT 약 6만 4천
화면	트웬티퍼스트 (21st MCP) `21st-dev/magic-mcp`	버튼 · 메뉴 같은 화면 부품을 찾아 코드를 가져온다	ISC 약 6천 4백
보안	스트릭스 `usestrix/strix`	내 앱을 실제로 공격해 보고 구멍을 찾는다	Apache 2.0 약 7만 4천

깔기 전에 알 것 셋

- **그냥 깔리는 것은 하나뿐입니다.** 에이전트 스킬스는 키가 없습니다. 컨텍스트 세븐과 트웬티퍼스트는 무료 키를 받고, 스트릭스는 도커와 LLM 키가 있어야 돕니다.
- **스트릭스는 플러그인이 아니라 터미널 도구입니다.** 터미널에서 따로 돌고, 클로드코드에서 부르려면 스킬을 한 줄 더 깝니다.
- **스트릭스는 내 앱에만 씁니다.** 실제로 공격을 보내는 도구입니다.

> 별 수 · 라이선스는 2026-11-11 깃허브 기준이고, 명령은 각 저장소 README 원문 그대로입니다. 이 자료를 만들면서 넷을 직접 깔아 돌리지는 않았습니다. 그래서 화면 결과가 아니라 README 에 적힌 것만 적었습니다.

준비물

키 받는 자리와 무료로 되는 범위

넷 다 오픈소스입니다. 다만 「무료」의 뜻이 도구마다 다릅니다.

도구	필요한 것	키 받는 자리	무료로 되는 범위 (README 기준)
에이전트 스킬스	클로드코드 · Git	키 없음	전부. 스킬은 글로 된 작업 순서라 따로 드는 돈이 없다
컨텍스트 세븐	Node.js 22.13 이상	`context7.com/dashboard` (설치 명령이 로그인 뒤 키를 만들어 준다)	키는 무료. README 는 키를 「권장」이라고 적고, 키가 있으면 한도가 더 높다고 적는다
트웬티퍼스트	Node.js	`21st.dev/mcp`	키는 무료. 부품 찾기(`search`) · 가져오기 (`get_component`)는 키만으로. AI 로 새로 만들기 (`generate`)는 AI 사용이 켜진 계정만. 도구 목록에 유료 코드 가져오기도 있다
스트릭스	도커(켜진 상태) · LLM 키	OpenAI · Anthropic · Google 등 쓰는 LLM 회사	도구는 무료. **LLM 사용료는 내가 낸다.** 점검 한 번에 모델을 여러 번 부른다

먼저 확인할 것

복사해서 그대로 쓰세요

```
claude --version
node --version
docker --version
```

확인	기준
`node --version`	22.13 이상이어야 컨텍스트 세븐 설치 명령(`ctx7`)이 돈다 (README)
`docker --version`	스트릭스를 쓸 때만. 버전이 나와도 도커 앱이 켜져 있어야 한다

> 트웬티퍼스트는 옛 「Magic MCP」가 이름을 바꾼 것입니다. README 는 옛 Magic 키가 모두 초기화됐다고 적습니다. 예전에 받은 키가 있어도 `21st.dev/mcp` 에서 새로 받으세요.

① 기획 · 설치

에이전트 스킬스, 클로드코드 안에서 두 줄

클로드코드를 열고 대화창에 한 줄씩 넣습니다.

복사해서 그대로 쓰세요

```
/plugin marketplace add addyosmani/agent-skills
/plugin install agent-skills@addy-agent-skills
```

첫 줄이 「어디서 받을지」를 등록하고, 둘째 줄이 실제로 깝니다. 깔린 뒤 클로드코드를 새로 열면 명령이 보입니다.

첫 줄에서 SSH 오류가 나면

README 의 안내입니다. 깃허브에 SSH 키를 등록하지 않았으면 첫 줄을 주소 전체로 바꿉니다.

복사해서 그대로 쓰세요

```
/plugin marketplace add https://github.com/addyosmani/agent-skills.git
/plugin install agent-skills@addy-agent-skills
```

그래도 `Permission denied (publickey)` 가 나오면 터미널에서 한 번만 아래를 넣습니다(README).

복사해서 그대로 쓰세요

```
git config --global url."https://github.com/".insteadOf git@github.com:
```

클로드코드 말고 다른 도구에도 넣고 싶으면

복사해서 그대로 쓰세요

```
npx skills add addyosmani/agent-skills
npx skills add addyosmani/agent-skills --list
```

첫 줄은 스킬 25개를 전부, 둘째 줄은 깔기 전에 목록만 봅니다. README 는 클로드코드에는 위의 플러그인 방식을 권합니다.

> 스킬 하나만 골라 깔면(`--skill 이름`) 저장소의 공용 체크리스트 폴더가 같이 오지 않습니다. 스킬은 돌지만 일부 참고 파일 경로가 비어 있다고 README 가 적습니다.

① 기획 · 쓰기

명령 아홉 개와 이름이 겹칠 때

명령은 모두 아홉 개입니다. 명령마다 맞는 스킬이 알아서 켜집니다.

하는 일	명령	원칙 (README)
무엇을 만들지 정한다	**/spec**	코드보다 명세 먼저
어떻게 만들지 나눈다	**/plan**	작고 쪼개진 작업
조금씩 만든다	**/build**	한 번에 한 조각
되는지 증명한다	**/test**	테스트가 증거
품질 기준을 정한다	**/constraints**	한 번 정하고 어디서나 지킨다
합치기 전에 검토한다	**/review**	코드 건강을 올린다
웹 속도를 점검한다	**/webperf**	고치기 전에 잰다
코드를 단순하게 한다	**/code-simplify**	재치보다 명료함
출시한다	**/ship**	빠른 것이 안전하다

이름이 겹칠 때

클로드코드에 원래 있는 명령과 이름이 같으면(`/review` 가 그렇습니다) 앞에 플러그인 이름을 붙입니다.

복사해서 그대로 쓰세요

```
/agent-skills:review
```

명령을 안 쳐도 켜집니다

README 는 하는 일에 따라 스킬이 스스로 켜진다고 적습니다. API 를 설계하면 `api-and-interface-design`, 화면을 만들면 `frontend-ui-engineering` 이 켜지는 식입니다.

> `/build auto` 는 계획을 한 번 승인받은 뒤 모든 작업을 이어서 만듭니다. 작업 사이에 사람이 넘겨주는 단계만 빠지고, 작업마다 테스트하고 따로 커밋하는 것은 그대로라고 README 가 적습니다.

② 문서

컨텍스트 세븐, 터미널에서 한 줄

클로드코드 안이 아니라 터미널에서 칩니다. Node.js 22.13 이상이 필요합니다.

복사해서 그대로 쓰세요

```
npx ctx7 setup --claude
```

README 에 적힌 이 명령이 하는 일은 셋입니다. 브라우저 로그인(OAuth) → API 키 만들기 → 클로드코드에 맞는 스킬 설치. 도중에 방식을 고르라고 묻습니다.

방식	무엇이 깔리나 (README)
CLI + Skills	클로드가 `ctx7` 명령으로 문서를 가져오게 안내하는 스킬. MCP 가 필요 없다
MCP	컨텍스트 세븐 MCP 서버를 등록해 클로드가 문서 도구를 직접 부른다

쓰는 법

부탁 끝에 `use context7` 을 붙입니다. 설치 명령으로 깔았으면 라이브러리 질문에 스킬이 알아서 켜지도록 설정됩니다(README).

복사해서 그대로 쓰세요

```
Next.js 16 에서 로그인 안 한 사람을 /login 으로 보내는 코드를 짜 줘. use context7
```

버전을 문장에 적으면 그 버전 문서를 찾습니다. 어떤 라이브러리인지 정확히 알면 ID 를 적어 찾는 단계를 건너뜁니다.

복사해서 그대로 쓰세요

```
Supabase 로 기본 로그인을 만들어 줘. use library /supabase/supabase for API and docs.
```

터미널에서 직접 볼 때

복사해서 그대로 쓰세요

```
ctx7 library next.js "app router"
ctx7 docs /vercel/next.js "dynamic route params"
```

첫 줄은 이름으로 라이브러리 ID 를 찾고, 둘째 줄은 그 ID 로 문서를 가져옵니다. 형식은 README 의 `ctx7 library <name> <query>` · `ctx7 docs <libraryId> <query>` 이고, 따옴표 안 검색어는 예시입니다. 문서는 각 프로젝트가 올린 것이라 정확성을 보장하지 않는다고 README 가 적습니다.

③ 화면

트웬티퍼스트, 키를 받고 한 줄

먼저 `21st.dev/mcp` 에서 무료 키를 받습니다. 그다음 터미널에서 한 줄입니다.

복사해서 그대로 쓰세요

```
npx @21st-dev/cli@latest init --client claude
```

`--client` 뒤에 쓰는 도구 이름을 적습니다. README 가 적은 값은 `cursor` · `claude` · `vscode` · `windsurf` · `codex` 입니다.

플러그인으로 깔 수도 있습니다

복사해서 그대로 쓰세요

```
claude plugin marketplace add 21st-dev/magic-mcp
/plugin install 21st
```

첫 줄은 터미널, 둘째 줄은 클로드코드 안입니다. 이 방식은 키를 `API_KEY_21ST` 라는 환경 변수에 넣어 둬야 합니다 (README).

키만으로 되는 것, 따로 켜야 하는 것

도구 이름	하는 일	조건 (README)
`search`	부품 찾기	키만 있으면 된다
`get_component`	부품 코드 가져오기	키만 있으면 된다
`generate`	AI 로 새로 만들기	AI 사용이 켜진 계정에만 목록에 보인다
`iterate_generation`	만든 것 다듬기	위와 같다

AI 사용이 꺼져 있으면 README 는 이렇게 쓰라고 적습니다. 찾아서 가져온 뒤, 고치는 것은 내 코딩 에이전트(클로드)에게 맡긴다.

복사해서 그대로 쓰세요

```
21st 에서 가격표(pricing) 부품을 찾아서 세 개만 보여 줘. 고른 것을 가져와서 우리 색에 맞게 고쳐 줘.
```

> `ai_subscription_required` 가 나오면 AI 사용이 꺼진 계정에서 만들기를 부른 것입니다. README 는 켜기 전에는 다시 시도하지 말라고 적습니다. 예전의 `/ui` · `/21` 같은 호출 낱말은 이제 쓰지 않고 말로 부탁합니다.

④ 보안 · 준비

스트릭스 돌리기 전 준비 셋

스트릭스는 내 앱을 실제로 돌려 보고 공격해서 구멍을 찾는 터미널 도구입니다. 돌리기 전에 셋을 갖춥니다.

1 · 도커를 켠다

README 준비물 첫 줄이 「Docker (running)」입니다. 깔려 있는 것으로는 부족하고 켜져 있어야 합니다. 첫 실행 때 점검용 도커 이미지를 알아서 내려받습니다.

2 · LLM 키를 넣는다

OpenAI · Anthropic · Google 등 지원하는 곳의 API 키가 필요합니다. 도구는 무료지만 **모델 사용료는 내 키에서 나갑니다.** 처음에는 작은 폴더 하나로 시험하세요.

3 · 내 앱에만 쓴다

README 맨 끝 경고입니다. 내가 가진 시스템이거나 **글로 된 명시적 허락**을 받은 대상에만 돌립니다. 허락 없는 점검은 대부분의 나라에서 불법이고 책임은 돌린 사람에게 있습니다.

설치와 키 넣기 (README 원문)

복사해서 그대로 쓰세요

```
curl -sSL https://strix.ai/install | bash
export STRIX_LLM="vercel_ai_gateway/zai/glm-5.3"
export LLM_API_KEY="your-api-key"
```

줄	뜻
1	스트릭스를 깐다. 인터넷에서 받은 스크립트를 바로 실행하는 형태이니 주소가 `strix.ai` 인지 확인한다
2	쓸 모델. 위 값은 README 의 예시(Vercel AI Gateway)다. README 는 OpenRouter 예시 `openrouter/z-ai/glm-5.3` 도 적는다. 내가 키를 받은 곳에 맞춰 적는다
3	그 회사에서 받은 키. `your-api-key` 자리에 넣는다

> 한 번 넣은 설정은 `~/.strix/cli-config.json` 에 저장돼 다음부터 다시 넣지 않습니다(README). 키가 그 파일에 남는다는 뜻이니 남과 같이 쓰는 컴퓨터에서는 조심하세요. ChatGPT Plus · Pro 구독으로 돌리는 길(`strix auth login chatgpt`)도 README 에 있습니다.

④ 보안 · 실행

돌리기, 결과 보기, 클로드에 붙이기

내 프로젝트 폴더에서 한 줄입니다.

복사해서 그대로 쓰세요

strix --target ./app-directory

`./app-directory` 자리에 점검할 폴더를 적습니다. 지금 폴더면 `./` 입니다.

대상	명령 (README)
내 컴퓨터의 코드 폴더	**`strix --target ./app-directory`**
내 깃허브 저장소	**`strix --target https://github.com/org/repo`**
내가 띄운 사이트	**`strix --target https://your-app.com`**

결과는 어디에

결과는 `strix_runs/<실행 이름>` 폴더에 저장됩니다. 화면으로 보려면 아래 한 줄입니다. 가장 최근 결과가 내 컴퓨터 안에서만 열립니다(README).

복사해서 그대로 쓰세요

strix view

클로드에 붙이기

스트릭스 본체는 터미널 도구입니다. 클로드코드에서 「점검해 줘」로 부르려면 스킬을 따로 깝니다.

복사해서 그대로 쓰세요

npx skills add usestrix/strix

README 는 이 한 줄이 스킬 아홉 개를 깐다고 적습니다. 점검 돌리기, 찾은 구멍 고치기, CI 점검 설정에 쓰는 스킬입니다. 스킬은 부르는 방법만 알려 주므로 위의 본체 설치와 준비 셋은 그대로 필요합니다.

> 오픈소스판이 확실히 하는 일은 「찾기」와 「결과 저장 · 고치는 방법 안내」입니다. 버튼 한 번으로 고친 코드를 올려 주는 기능(one-click autofix)은 README 가 유료 서비스인 Strix Cloud 쪽에 적습니다. 도커와 키 없이 돌리는 `strix cloud` 는 코드를 그 서비스로 올리는 방식입니다.

순서대로 쓰기

넷을 이어 쓰는 요청 문장

기능 하나를 만들 때 넷을 순서대로 부르는 예시입니다. 대괄호 안만 내 것으로 바꿉니다.

순서	어디서	무엇을 넣나
1 기획	클로드코드	`/spec` 뒤에 만들 것을 한 문장으로
2 계획	클로드코드	`/plan`
3 문서	클로드코드	만들기 부탁 끝에 `use context7`
4 화면	클로드코드	「21st 에서 찾아 줘」
5 검토	클로드코드	`/agent-skills:review`
6 보안	터미널	`strix --target ./`

복사해서 그대로 쓰세요

/spec [회원이 예약을 잡고 취소하는 화면]을 만들고 싶어. 모르는 건 하나씩 물어봐 줘.

복사해서 그대로 쓰세요

/plan

복사해서 그대로 쓰세요

/build 첫 번째 작업부터 해 줘. [Next.js] 문법은 지금 버전 문서를 확인하고 써 줘. use context7

복사해서 그대로 쓰세요

21st 에서 [예약 달력] 부품을 찾아서 세 개만 보여 줘. 고른 것을 가져와서 지금 화면에 맞게 고쳐 줘.

복사해서 그대로 쓰세요

/agent-skills:review

복사해서 그대로 쓰세요

strix --target ./

> 명령과 `use context7` 은 README 원문이고, 나머지 문장은 이 자료가 지은 예시입니다. 마지막 줄만 터미널에서 칩니다. 스트릭스는 모델 사용료가 드니 매번이 아니라 출시 전처럼 필요한 때에 돌리세요.

지우기 · 막히면

지우는 법과 자주 막히는 곳

넷 다 따로 깔리므로 따로 지웁니다.

도구	지우는 명령 (터미널)	근거
에이전트 스킬스	**`claude plugin uninstall agent-skills@addy-agent-skills`** **`claude plugin marketplace remove addy-agent-skills`**	클로드코드 도움말 (저장소 README 에는 없음)
컨텍스트 세븐	**`npx ctx7 remove`**	README. `npm install -g ctx7` 로 깔았으면 `npm uninstall -g ctx7` 도
트웬티퍼스트	**`claude mcp list`** **`claude mcp remove [이름]`**	클로드코드 도움말. 첫 줄로 등록된 이름을 본 뒤 지운다. 플러그인으로 깔았으면 `claude plugin uninstall 21st`
스트릭스 스킬	**`npx skills list`** **`npx skills remove [스킬 이름]`**	skills CLI README
스트릭스 본체	README 에 지우는 명령이 없다. `which strix` 로 실행 파일 자리를 확인한다. 설정과 키는 `~/.strix/` , 결과는 `strix_runs/` 에 남는다	

증상 → 할 일

증상	할 일
플러그인 추가에서 SSH · publickey 오류	주소 전체(`https://github.com/addyosmani/agent-skills.git`)로 다시 추가한다
`/review` 가 다른 것을 실행한다	클로드코드 기본 명령과 이름이 같다. `/agent-skills:review` 로 부른다
`npx ctx7` 이 실행되지 않는다	`node --version` 이 22.13 이상인지 본다
트웬티퍼스트 옛 키가 안 된다	옛 Magic 키는 모두 초기화됐다. `21st.dev/mcp` 에서 새로 받는다
`generate` 가 목록에 없다	AI 사용이 꺼진 계정이다. `search` · `get_component` 로 찾아 가져온다
스트릭스가 시작하지 못한다	도커 앱이 켜져 있는지, `LLM_API_KEY` 를 넣었는지 본다

더 묻고 싶으면 네이버 카페 「바이브메이커」 자료실 글 댓글로.