Eigenflow

GUIDE · CLAUDE CODE

Cinq outils GitHub gratuits pour Claude

FreeLLMAPI, Lemo Opuscar, Ponytail, Agent Reach et SkillSpector : ce qu'ils font, l'installation, la licence et les risques, vérifiés le 14 novembre 2026 et relus le 23.

PAR

Abyane Nabeebaccus

INSTAGRAM

@abyane_nb

ÉDITION

novembre 2026

Tu as commenté FACTURE, voici le guide. Cinq dépôts GitHub gratuits qui tournent avec Claude ou à côté : FreeLLMAPI, Lemo Opuscar, Ponytail, Agent Reach et SkillSpector. Pour chacun, ce qu'il fait vraiment, à qui il sert, l'installation tirée de son README, ce qui est gratuit et ce qui ne l'est pas, les limites, les risques et la licence. Puis la marche à suivre avant d'installer une skill, avec SkillSpector.

Dépôts, documentation et conditions des plateformes relus le 14 novembre 2026. On n'a rien installé ni lancé : les commandes viennent des README, copiées telles quelles, et on le signale quand une variante est de nous. On n'a aucun lien avec leurs auteurs, ni avec NVIDIA, ni avec Anthropic.

À savoir avant de lire

- Les cinq outils sont gratuits. Claude Code ne l'est pas : Lemo Opuscar et Ponytail tournent dedans, et il faut un abonnement Claude payant, dès 20 $ HT par mois (Pro).
- Aucun n'est fait par Anthropic. SkillSpector vient de NVIDIA, les quatre autres de développeurs indépendants.
- Les 10,1 milliards de tokens de FreeLLMAPI additionnent les offres gratuites de 87 fournisseurs. Tu n'as que celles où tu ouvres un compte. Ce sont d'autres modèles que Claude qui répondent, et le dépôt réserve l'outil à un usage perso.
- Ponytail : 53 % de code en moins et 45 % de tokens écrits en moins, d'après le banc d'essai de son auteur. Sur la facture, c'est 26 % de moins.
- Agent Reach lit X et Reddit avec ton compte connecté. Leurs conditions interdisent la collecte automatisée sans accord : prends un compte secondaire.
- Lemo Opuscar : les films que tu fais sont à toi, y compris pour un client. Ses fiches de style, elles, ne se revendent pas.
- L'étude « une skill sur quatre » porte sur 31 132 skills collectées en décembre 2025. Elle vient de chercheurs universitaires, NVIDIA la cite.

---

1. FreeLLMAPI, des modèles gratuits derrière une seule adresse

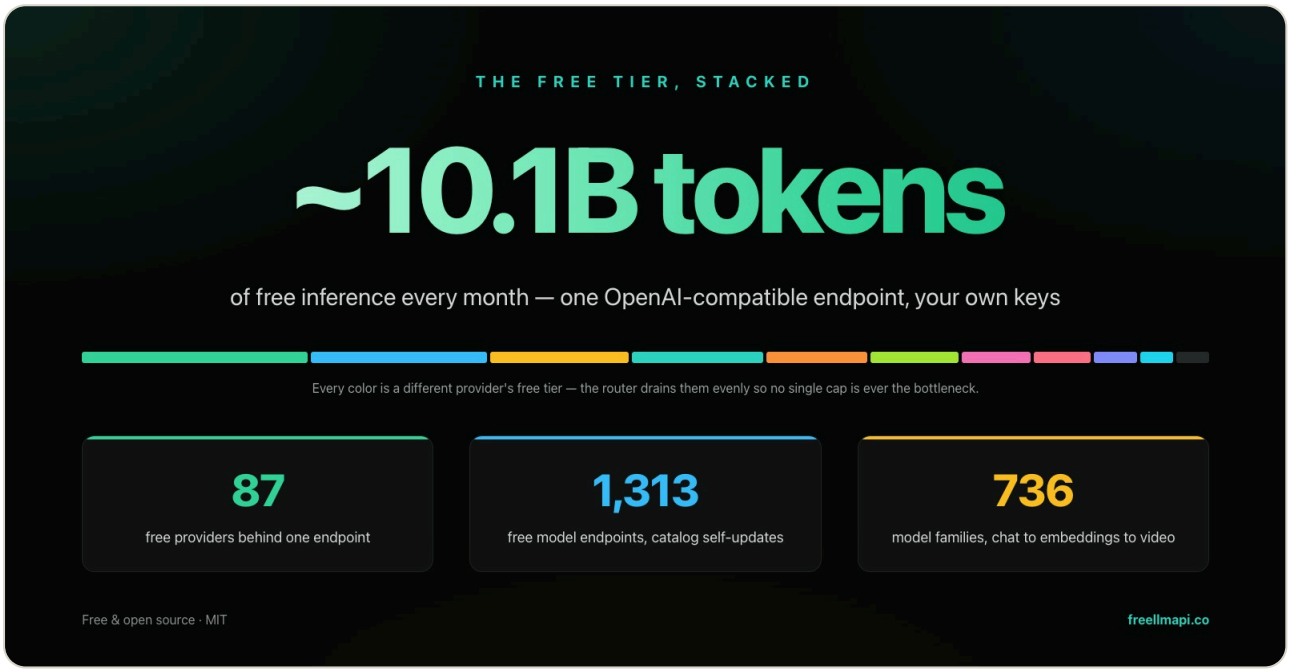
Le dépôt : github.com/tashfeenahmed/freellmapi. Licence MIT, 52 069 étoiles le 14 novembre 2026, version v0.16.29 sortie le jour même.

Ce qu'il fait

FreeLLMAPI installe un petit serveur sur ta machine. Tu y ranges tes clés gratuites (Groq, Google AI Studio, OpenRouter, Mistral et des dizaines d'autres) et il les présente derrière une seule adresse, au format d'OpenAI et à celui d'Anthropic. Claude Code peut donc lui parler comme il parle à Anthropic.

À chaque requête, le routeur prend le modèle le mieux classé de ta liste qui a encore du quota. Si le fournisseur répond « trop de requêtes » ou tombe en erreur, il passe au modèle suivant. L'en-tête X-Routed-Via de chaque réponse dit qui a répondu.

Les 10,1 milliards de tokens par mois, c'est la somme des quotas gratuits de son catalogue : 87 fournisseurs, 736 familles de modèles, 1 313 points d'accès. Le routeur n'utilise que les fournisseurs où tu as mis une clé, plus quelques-uns qui n'en demandent pas. Avec trois clés, tu as une petite partie de ce total.



L'image du README de FreeLLMAPI le 14 novembre 2026 : environ 10,1 milliards de tokens gratuits par mois, 87 fournisseurs, 1 313 points d'accès, 736 familles de modèles

À qui il sert

À qui veut essayer des modèles gratuits dans ses outils habituels (Claude Code, Codex, Cursor, Cline, Aider), sur des projets perso. Pas au travail client : la description du dépôt finit par « Personal experimentation only ».

L'installer

Trois voies d'après le README.

L'appli de bureau : sur la page Releases, prends le .dmg arm64 pour un Mac Apple Silicon, le x64 pour un Mac Intel (macOS 12 au minimum) ou l'installeur .exe pour Windows. Pas de compte à créer. Le tableau de bord s'ouvre depuis l'icône de la barre des menus (Open Dashboard), et le menu copie ta clé unifiée.

Docker, en une ligne (Docker installé et démarré) :

À COLLER

```
curl -fsSL https://freellmapi.co/install.sh | bash
```

Cette ligne exécute un script téléchargé à la volée. Le README propose de le lire avant. Notre variante, pour le lire d'abord (ici dans TextEdit) :

À COLLER

```
curl -fsSL https://freellmapi.co/install.sh -o install.sh
open -e install.sh
bash install.sh
```

Ouvre ensuite http://localhost:3001, crée le compte du tableau de bord et ajoute tes clés sur la page Keys. Ta clé unifiée, qui commence par freellmapi-, s'affiche en haut de cette page.

La troisième voie, depuis le code source, est dans la doc d'installation.

Le brancher sur Claude Code

La commande la plus prudente : `launch` ouvre Claude Code branché sur le routeur, pour cette session seulement. D'après le README de la commande, elle n'écrit aucune clé sur le disque.

À COLLER

```
npx freellmapi launch --url http://localhost:3001
```

Sans l'option `--api-key`, elle te demande ta clé unifiée sans l'afficher. Il faut Node.js 20.18 ou plus récent. Avec l'appli de bureau, remplace l'adresse par celle que donne son menu.

Le README met aussi en avant `setup-claude`, qui écrit ses réglages dans la configuration de Claude Code après une copie de sauvegarde. Sans option, c'est le fichier `~/.claude/settings.json` (d'après le code de la commande), et toutes tes sessions passent alors par le routeur. Avec `--profile`, elle écrit un profil à part. Regarde d'abord ce qu'elle écrirait, sans rien modifier :

À COLLER

```
npx freellmapi setup-claude --profile gratuit --url http://localhost:3001 --dry-run
```

Relance-la sans `--dry-run` pour l'écrire. Ce profil s'ouvre ensuite ainsi :

À COLLER

```
env CLAUDE_CONFIG_DIR="$HOME/.claude/profiles/gratuit" claude
```

Ce qui est gratuit, ce qui ne l'est pas

- Le routeur : gratuit, sous MIT. Le README promet qu'il le restera.
- Premium : 19 $ par an ou 49 $ une fois. Il te donne les nouveaux modèles du catalogue le jour même. Sans lui, chacun arrive 30 jours plus tard ; le README chiffre le retard de la version gratuite à environ 303 modèles.
- Les clés : gratuites chez la plupart des fournisseurs, dans la limite de leurs quotas. Les offres changent sans prévenir : le 7 novembre, Cerebras, encore rangé parmi les gratuits, demandait un moyen de paiement pour ses crédits d'essai.
- Branché sur FreeLLMAPI, Claude Code ne puise plus dans ton abonnement Claude. C'est logique : ce ne sont plus des modèles Claude qui répondent.

Limites et risques

- Le README l'écrit : « This project is for personal experimentation and learning, not production. » Avant de livrer quoi que ce soit, il conseille une API payante.
- Les meilleurs modèles ont les plus petits quotas. Quand ils sont épuisés, le routeur descend vers des modèles plus faibles jusqu'à minuit UTC, soit 1 h du matin à Paris en hiver.
- Latence variable, aucune garantie de service.
- Tes requêtes partent chez les fournisseurs que tu branches, sous leurs conditions. Sur l'offre gratuite de l'API Gemini, Google se sert de ce que tu envoies et des réponses pour améliorer ses produits, et des relecteurs humains peuvent les lire. Sa consigne : « Do not submit sensitive, confidential, or personal information to the Unpaid Services. » Pas de code ni de données client, donc.
- La revue des conditions faite par le dépôt en mai 2026 range Cohere à éviter (usage perso interdit) et demande la prudence avec Google Gemini, NVIDIA NIM (« evaluation only, not production ») et Z.ai. Ses règles : un compte par fournisseur, pas de revente, un serveur pour toi seul.
- Anthropic ne prend pas en charge ce montage. Sa doc : « Anthropic doesn't endorse, maintain, or audit third-party gateway products, and doesn't support routing Claude Code to non-Claude models through any gateway. »

Licence

MIT, usage commercial permis pour le code du routeur. Les conditions de chaque fournisseur s'appliquent en plus, et ce sont elles qui limitent l'usage.

---

2. Lemo Opuscar, 99 styles de films faits en code par Claude

Le dépôt : github.com/lemomo-ai/lemo-opuscar. 15 915 étoiles le 14 novembre 2026, version 2.0.0 du 12 novembre. La galerie des styles : lemomo-ai.github.io/lemo-opuscar.

Ce qu'il fait

Une bibliothèque de 99 styles de films pour Claude Code, rangés en 15 catégories. Quelques noms : Ukiyo-e, Watercolor, Neon Tubes, LiDAR Survey, Swiss Motion Graphics. Chaque style a

une fiche (`STYLE.md`) et un film de démo. Tu choisis un style, tu donnes ton sujet, et Claude écrit tout le film en code : images en canvas et WebGL rendues image par image, musique tirée de banques de sons libres, voix de synthèse. Aucun modèle vidéo, aucune banque d'images.

Avant de commencer, il te pose ses questions une seule fois : ce qu'il ne peut pas décider sur ton sujet, si tu as ta voix, ta musique ou tes images, et si tu veux voir un storyboard d'abord.

Tu peux aussi te faire un style à toi, à partir d'un style de la bibliothèque ou d'une page blanche. Il se range dans `~/lemo-my-styles/` et survit aux mises à jour.



Le haut de la page GitHub de Lemo Opuscar le 14 novembre 2026 : « A toolbox of film styles made in code for Claude Code: 99 ready to use », 15,9 k étoiles, 2,0 k forks

À qui il sert

À qui veut des films explicatifs, des annonces de lancement ou des clips sans tourner ni générer de vidéo. Compte 30 à 60 minutes de travail de l'agent par film, d'après le README.

L'installer

Il te faut Node 20 ou plus récent, ffmpeg 7 ou plus récent et Python 3.11 ou plus récent (ou uv). L'agent installe le reste. Dans ton terminal, les deux commandes du README :

```
À COLLER
claude plugin marketplace add lemomo-ai/lemo-opuscar
```

```
À COLLER
claude plugin install lemo-opuscar@lemolab
```

Au premier usage, la skill télécharge guides, outils et fiches de style dans `~/lemo-opuscar`. Chaque film se range dans le dossier d'où tu as lancé Claude Code.

Le README conseille de régler l'effort au niveau haut. Dans Claude Code :

À COLLER

```
/effort high
```

Puis tape la commande seule : elle te demande la langue, puis ce que tu veux faire.

À COLLER

```
/lemo-opuscar:lemo-opuscar
```

Ou une phrase directe, comme dans l'exemple du README :

À COLLER

```
Make a film in the Neon Tubes style about the coffee farm my family runs.
```

Le film sort par défaut en 1920×1080, à l'horizontale. Pour Instagram ou TikTok, demande le vertical. Notre adaptation en français, non testée :

À COLLER

```
Fais un film de 40 secondes au format vertical 1080×1920, dans le style
Watercolor, sur la boulangerie de mon quartier. Voix off en français.
```

Pour désinstaller, puis supprimer le dossier `~/lemo-opuscar` :

À COLLER

```
claude plugin uninstall lemo-opuscar@lemolab
```

Ce qui est gratuit, ce qui ne l'est pas

- La bibliothèque est gratuite.
- Chaque film puise dans ton abonnement Claude, et le README parle d'une bonne quantité de tokens par film.
- Trois styles (Manga Panels, Text Flash Cuts, Comic History Cartoon) demandent un générateur d'images à toi, par exemple Codex CLI connecté à un compte qui génère des images. Les 96 autres s'en passent.

Limites et risques

- Fait pour Claude : Fable, Opus et Sonnet marchent. D'après l'auteur, les autres modèles risquent de mal reproduire les styles.
- La skill installe des paquets (npm, Python) et télécharge des banques de sons au fil des films. C'est du code tiers qui tourne sur ta machine (partie 6).
- Clawd, le petit personnage orange de certains films de démo, appartient à Anthropic. Aucune licence n'est donnée dessus.

Licence : ce que tu peux faire, ce que tu ne peux pas

Depuis la version 2.0.0 du 12 novembre, la licence a plusieurs parties.

- Le code : MIT.
- Les fiches et les guides (tous les fichiers Markdown, dont chaque `STYLE.md`) : CC BY-NC 4.0, donc pas d'usage commercial en principe. L'auteur ajoute une permission : toi et ceux qui travaillent avec toi sur un film (salariés, prestataires, client) pouvez les lire, les suivre, les copier et les adapter pour faire vos films, films commerciaux et films clients compris.
- Les films que tu fais : à toi, usage commercial compris, sans crédit obligatoire.
- Les films de démo et leurs images : CC BY 4.0, réutilisables avec crédit, sauf Clawd, les interfaces et logos de tiers, et six pistes de voix à ne pas extraire.

Ce qui demande une licence commerciale, à demander à lemomo.studio@gmail.com : vendre les fiches, ou les mettre, même en partie ou adaptées, dans un produit ou un service exploité commercialement, payant ou non. L'auteur explique ce changement : sa bibliothèque était reconditionnée et vendue dans des produits payants.

En clair : faire des films pour toi ou pour un client, oui. Construire une appli de vidéo sur ses fiches, non, sans son accord. Les versions publiées avant la 2.0.0 gardent leur licence d'origine.

---

3. Ponytail, Claude écrit moins de code

Le dépôt : github.com/DietrichGebert/ponytail. Licence MIT, 190 613 étoiles le 14 novembre 2026, version 5.3.3 du 29 octobre.

Ce qu'il fait

Une skill qui fait penser ton agent comme « the laziest senior dev in the room ». Avant d'écrire du code, Claude lit ce que le changement touche, puis s'arrête au premier barreau qui tient : est-ce que ça doit exister, est-ce déjà dans le projet, la bibliothèque standard le fait-elle, la plateforme le fait-elle, une dépendance déjà installée, une seule ligne suffit-elle. Seulement ensuite, le minimum qui marche, avec un petit test s'il y a de la logique.

Ce qu'il ne coupe jamais, d'après le README : la validation des entrées, la gestion des pertes de données, la sécurité et l'accessibilité. Chaque réponse finit par ce qui a été sauté ou pas vérifié.

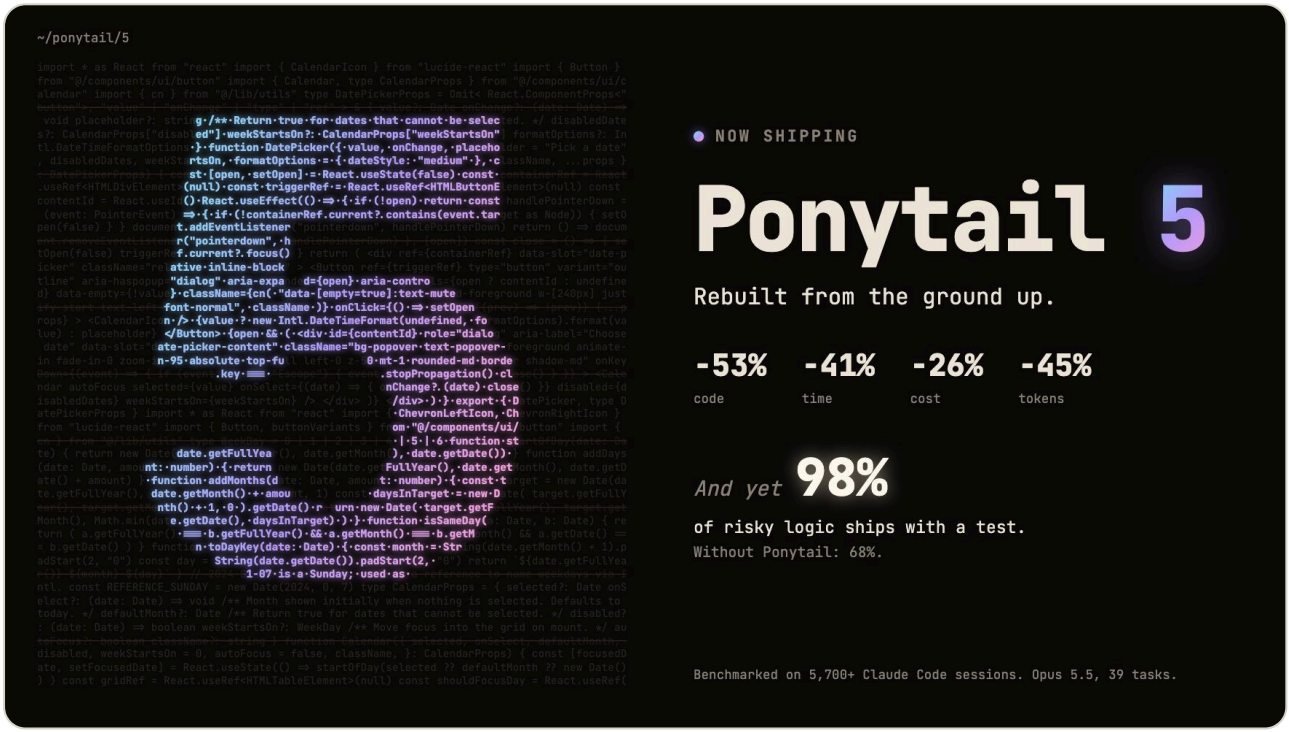
L'exemple du README : un sélecteur de date. Sans Ponytail, 335 lignes et un calendrier fait main. Avec, un fichier de 10 lignes qui réutilise le composant Input du projet en type date : le navigateur fournit le calendrier.

Il ajoute des commandes : `/ponytail-review` relit ton changement comme le senior qu'on réveille quand ça casse, `/ponytail-audit` passe tout le dépôt en revue, `/ponytail lite`, `full`, `ultra` ou `off` règle l'intensité.

Les chiffres, et d'où ils viennent

Le banc d'essai de l'auteur, daté du 7 octobre 2026 : Claude Code, Opus 5.5, 39 tâches dont 12 dans un vrai dépôt FastAPI et React, 5 passages par tâche, 585 sessions. Contre le même agent sans skill :

- 53 % de lignes de code en moins ;
- 45 % de tokens écrits en moins (les tokens de sortie) ;
- 26 % de coût en moins au prix de l'API, 0,142 $ par session au lieu de 0,200 $;
- 41 % de temps en moins.



L'image du README de Ponytail 5 le 14 novembre 2026 : moins 53 % de code, moins 41 % de temps, moins 26 % de coût, moins 45 % de tokens, et 98 % de la logique risquée livrée avec un test

Ce que ces chiffres ne disent pas :

- C'est l'auteur qui mesure, sur un seul modèle, avec Bash coupé pour tous les agents.
- Les 45 % portent sur ce que Claude écrit. Le coût total baisse de 26 %.
- Moins de code n'a rien coûté en justesse, et n'en a pas ajouté : 87 contrôles cachés réussis sur 90 avec Ponytail, 86 sur 90 sans.
- Les réponses sont 30 % plus courtes. Un juge à l'aveugle a même un peu préféré celles sans skill, un écart que l'auteur donne comme non significatif.

L'installer

Dans Claude Code, en deux messages séparés, comme le dit le README :

À COLLER

```
/plugin marketplace add DietrichGebert/ponytail
```

À COLLER

```
/plugin install ponytail@ponytail
```

Pour Codex :

À COLLER

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

Puis ouvre `/hooks` dans Codex, accepte ses deux hooks et démarre un nouveau fil. Pour les autres agents (Cursor, Copilot, Gemini…), le pas à pas est dans INSTALL.md.

Ensuite il est actif à chaque session. Pour le couper :

À COLLER

```
/ponytail off
```

Ce qui est gratuit, ce qui ne l'est pas

Gratuit, sous MIT. Il tourne dans Claude Code, donc avec ton abonnement, ou dans un autre agent. Il réduit ce que tu consommes, sans remplacer de service payant.

Limites et risques

- Installe-le seulement depuis DietrichGebert/ponytail sur GitHub ou @dietrichgebert/ponytail sur npm. L'auteur prévient qu'il ne livre jamais de fichiers .exe ni .dll : une copie qui en contient n'est pas la sienne.
- Le README dit « Ponytail is one prompt », mais le plugin Claude Code ajoute aussi des hooks, des scripts JavaScript lancés à chaque session, et une bande au-dessus de la saisie (coût, tokens, temps). Par défaut, tout reste sur ta machine, dans `~/.config/ponytail/band/`.
- La bande demande Claude Code 2.1.290 ou plus récent. Elle ne s'affiche pas avec `claude -p` ni dans le panneau de VS Code.

Licence

MIT, usage commercial permis.

4. Agent Reach, X, Reddit, YouTube et LinkedIn pour ton agent

Le dépôt : github.com/Panniantong/Agent-Reach. Licence MIT, 111 691 étoiles le 14 novembre 2026, version 1.6.1 du 1er novembre. Le README principal est en chinois, la version anglaise est dans docs/README_en.md.

Ce qu'il fait

Un outil en ligne de commande qui installe et teste, pour ton agent, de quoi lire les plateformes qu'il n'atteint pas seul. Il ne fait que lire : d'après le README, il ne publie rien, ne commente pas, ne like pas et n'envoie aucun message.

Ce qu'il lit, d'après sa table :

- GitHub, YouTube (infos, recherche, sous-titres) et le web : dès l'installation, sans compte.
- LinkedIn : les offres d'emploi et leur détail. Les offres publiques se lisent sans connexion.
- Reddit : recherche, posts, commentaires, une fois connecté.
- X : recherche, profils, fils, une fois connecté. Classé expérimental : il peut casser à tout moment.
- Et aussi Bilibili, V2EX, Xiaohongshu, Xueqiu, Facebook et Instagram.

Pour chaque action, il essaie une liste d'outils dans l'ordre et passe au suivant si l'un échoue. Pour lire une page web, par exemple, Jina Reader puis Exa.

À qui il sert

À qui fait de la veille ou de la recherche depuis Claude Code ou un autre agent : résumer une vidéo, lire un fil Reddit, suivre des offres d'emploi, fouiller GitHub.

L'installer

Le README propose une phrase à envoyer à ton agent, qui suit alors le guide d'installation du dépôt :

```
À COLLER

Install Agent Reach: https://raw.githubusercontent.com/Panniantong/Agent-Reach/main/docs/install.md
```

Les commandes que ce guide fait lancer. D'abord uv, s'il manque sur ton Mac :

```
À COLLER

curl -LsSf https://astral.sh/uv/install.sh | sh
```

Puis :

```
À COLLER

uv tool install https://github.com/Panniantong/Agent-
Reach/releases/latest/download/agent-reach.zip
agent-reach setup
```

Les canaux optionnels s'ajoutent un par un :

```
À COLLER

agent-reach add reddit
agent-reach add twitter
agent-reach add linkedin
```

Le paquet nommé agent-reach sur PyPI n'est pas ce projet. Installe seulement depuis l'adresse GitHub.

Le compte que tu lui donnes

Pour X et Reddit, il faut une session connectée. Au premier usage d'un site, Agent Reach copie la connexion de ce seul site depuis ton Chrome, sans les mots de passe, si l'extension OpenCLI y est branchée. En général, c'est ton compte principal.

Son README le dit : « platforms may restrict accounts used for automation ». Et les conditions des deux plateformes sont nettes :

- X, conditions en vigueur depuis le 9 octobre 2026 : « crawling or scraping the Services in any form, for any purpose without our prior written consent is expressly prohibited ». X peut suspendre un compte s'il estime que tu as enfreint ses conditions.
- Reddit, accord en vigueur depuis le 1er juillet 2026 : « scraping the Services without Reddit's prior written consent is prohibited ».

Le risque de restriction est réel, pas automatique, et personne ne le chiffre. Prends un compte secondaire, et connecte-le toi-même dans la fenêtre d'Agent Reach avant le premier usage :

```
À COLLER

agent-reach login twitter
agent-reach login reddit
```

D'après sa doc des sessions, une fois que tu t'es connecté toi-même, il ne recopie plus la connexion de Chrome. Pour effacer une connexion : `agent-reach logout twitter`.

Pour les canaux qui passent par son navigateur, il limite lui-même le rythme : deux requêtes à la fois au plus par plateforme, et 10 secondes entre deux recherches sur la même.

Ce qui est gratuit, ce qui ne l'est pas

Gratuit, sous MIT, sans API payante des plateformes. Seule la transcription d'une vidéo YouTube sans sous-titres demande une clé Groq ou OpenAI. Tape-la dans ton terminal, jamais dans le chat

:

```
À COLLER

agent-reach configure groq-key
```

Limites et risques

- Les plateformes changent souvent leurs protections : un canal peut casser du jour au lendemain, X et Instagram en premier.
- Ce qu'il lit part ensuite dans ton agent, donc chez le fournisseur de ton modèle.
- LinkedIn interdit aussi les logiciels de collecte dans ses conditions. Agent Reach ne lit que les offres publiques sans connexion, donc aucun compte n'est exposé.
- Tout ce qu'il installe vit dans `~/.agent-reach/`. Pour tout enlever, dis « Uninstall Agent Reach » à ton agent : il liste ce qu'il va supprimer et attend ton accord.

Licence

MIT, usage commercial permis. Les conditions de chaque plateforme s'appliquent en plus.

---

5. SkillSpector, le scanner de skills de NVIDIA

Le dépôt : github.com/NVIDIA/SkillSpector. Licence Apache-2.0, 25 971 étoiles le 14 novembre 2026, version 2.13.4 du 4 novembre. La doc de NVIDIA : Scan Agent Skills Before Installation.

Ce qu'il fait

Il analyse une skill avant que tu l'installes : un dossier, un fichier SKILL.md, un dépôt Git ou un zip. Il cherche 71 motifs en 17 catégories, dont l'injection de prompt, l'exfiltration de données, l'élévation de privilèges, les dépendances aux failles connues et l'empoisonnement des outils MCP.

Deux étages. D'abord une analyse statique rapide (motifs, structure du code Python, signatures YARA). Ensuite, si tu la branches, une analyse par un modèle de langage, qui compare ce que la skill dit faire et ce que son code semble faire. Il n'exécute jamais la skill.

Il rend une note de 0 à 100 et une recommandation :

- de 0 à 20 : risque LOW, recommandation SAFE ;
- de 21 à 50 : risque MEDIUM, recommandation CAUTION ;
- de 51 à 80 : risque HIGH, recommandation DO NOT INSTALL ;
- de 81 à 100 : risque CRITICAL, recommandation DO NOT INSTALL.

NVIDIA s'en sert pour vérifier ses propres skills avant de les publier.



Le haut de la page GitHub de SkillSpector le 14 novembre 2026, sous le compte NVIDIA : « Security scanner for AI agent skills », licence Apache-2.0, 26,0 k étoiles

L'étude derrière « une skill sur quatre »

Le README de SkillSpector cite une étude de chercheurs de Quantstamp et de cinq universités (Tianjin, Southern Cross, NTU à Singapour, UNSW, Griffith) : Agent Skills in the Wild, de Yi Liu et ses coauteurs, déposée sur arXiv le 15 janvier 2026. Ils ont collecté 42 447 skills sur deux places de marché, skills.rest et skillsmp.com, en décembre 2025, et en ont analysé 31 132 après dédoublonnage. On l'a vérifié sur l'article.

- 26,1 % ont au moins une vulnérabilité, soit 8 126 skills. Les auteurs placent le vrai taux entre 23 et 30 %.
- 5,2 % montrent des motifs graves qui suggèrent fortement une intention malveillante.
- Les skills qui embarquent des scripts sont 2,12 fois plus souvent vulnérables que celles faites seulement d'instructions.
- Leur détecteur se trompe parfois (précision de 86,7 %), et les skills de sécurité, dangereuses par nature, gonflent le total. Sans elles, on tombe à 24,8 %.

L'installer

Il te faut uv, et Python 3.12 ou 3.13 : le module YARA dont il dépend n'a pas encore de version précompilée pour Python 3.14, et l'installation peut alors échouer à la compilation (README relu le 23 novembre). La commande du README choisit 3.13 d'office :

```
À COLLER
uv tool install --python 3.13 git+https://github.com/NVIDIA/skillspector.git
```

Pour le mettre à jour plus tard : `uv tool update skillspector`. Le README prévient que l'installation télécharge d'autres logiciels open source, chacun sous sa licence.

Ce qui part chez un modèle, et ce qui reste chez toi

- Avec l'option `--no-llm`, l'analyse reste sur ta machine. Seuls les noms et versions des dépendances partent vers OSV.dev, la base publique des failles connues, même dans ce mode.
- Sans `--no-llm`, SkillSpector envoie le contenu des fichiers au fournisseur choisi avec `SKILLSPECTOR_PROVIDER`. Par défaut, celui de NVIDIA (build.nvidia.com), avec une clé `NVIDIA_INFERENCE_KEY`.
- Sans clé d'API : le fournisseur `claude_cli` passe par ta session Claude Code, et le contenu part alors chez Anthropic. Avec `ollama`, tout reste sur ta machine.
- La page de NVIDIA dit que l'analyse statique est le mode par défaut. Le README dit que l'analyse par modèle est active par défaut. Écris `--no-llm` quand tu veux un scan local, et le doute disparaît.

Ce qui est gratuit, ce qui ne l'est pas

Le scanner est gratuit, sous Apache-2.0, et l'analyse statique ne coûte rien. L'analyse par modèle consomme une clé payante, ton abonnement Claude ou la puissance de ta machine.

Limites

D'après son README :

- Il peut rater des motifs écrits dans une autre langue que l'anglais. Une skill en français est moins bien couverte.
- Il ne lit pas le texte dans les images, ni le code chiffré ou compilé.
- Il ne voit pas ce que fait la skill quand elle tourne.
- Il ne protège rien après l'installation : « It does not sandbox the host ».

Licence

Apache-2.0, usage commercial permis.

---

6. Avant d'installer une skill

La marche qu'on te conseille, d'après le README de SkillSpector et la page de NVIDIA. On ne l'a pas lancée nous-mêmes.

Étape 1. Vérifie d'où elle vient

Le bon dépôt, le bon auteur. Ponytail et Agent Reach préviennent eux-mêmes contre des copies : un paquet npm ou PyPI au même nom, une version qui livre des .exe.

Étape 2. Un premier scan, sur ta machine

Pour un dépôt GitHub, remplace auteur et skill par les vrais noms :

```
À COLLER

skillspector scan https://github.com/auteur/skill --no-llm
```

Pour une skill déjà téléchargée :

```
À COLLER

skillspector scan ./ma-skill/ --no-llm
```

Commence par les outils de ce guide :

```
À COLLER

skillspector scan https://github.com/DietrichGebert/ponytail --no-llm
skillspector scan https://github.com/Panniantong/Agent-Reach --no-llm
```

Le dépôt de Lemo Opuscar pèse environ 200 Mo d'après GitHub, probablement au-dessus de la limite de 100 Mio que SkillSpector applique à un dépôt cloné. Notre variante : clone-le, puis scanne le dossier de la skill.

```
À COLLER

git clone https://github.com/lemomo-ai/lemo-opuscar.git
skillspector scan ./lemo-opuscar/plugin/skills/lemo-opuscar/ --no-llm
```

Ce scan couvre la skill d'entrée. Les outils qu'elle télécharge ensuite viennent des dossiers `core/` et `tools/` du même dépôt : scanne-les aussi si tu veux tout voir.

Étape 3. Lis la recommandation

SAFE, tu peux continuer. CAUTION, lis les fichiers signalés avant d'installer. DO NOT INSTALL, n'installe pas. Le README propose la même règle : autoriser, avertir, bloquer.

Par défaut, la commande se termine de la même façon pour SAFE et pour CAUTION. Lis la recommandation dans le rapport, pas seulement le résultat de la commande.

Étape 4. Lis toi-même ce qui exécute du code

Les scripts de la skill : un dossier scripts, des hooks, des fichiers .py, .js ou .sh. Dans l'étude, ce sont eux qui rendent une skill 2,12 fois plus souvent vulnérable. Une bonne note ne garantit rien : le scanner a ses angles morts (partie 5).

Étape 5. Une passe par modèle, si tu as un doute

Avec ta session Claude Code, sans clé d'API :

```
À COLLER

export SKILLSPECTOR_PROVIDER=claude_cli
skillspector scan https://github.com/auteur/skill
```

Le contenu de la skill part alors chez Anthropic. Pour garder un rapport lisible, ajoute `--format markdown --output rapport.md`.

Étape 6, en option : le brancher dans Claude Code

SkillSpector peut tourner comme serveur MCP : Claude appelle alors le scan avant d'installer une skill. Il faut l'option mcp à l'installation :

```
À COLLER

uv tool install --python 3.13 --force 'skillspector[mcp] @
git+https://github.com/NVIDIA/skillspector.git'
claude mcp add skillspector -- skillspector mcp
```

Le README signale un blocage possible au démarrage dans ce mode (ticket n° 199 du dépôt).

---

7. Notre usage

- Agent Reach fait partie de notre veille depuis juillet 2026 : sous-titres YouTube, dépôts GitHub, pages web. En octobre, on l'a aussi branché sur Instagram avec notre compte principal, par OpenCLI, l'outil qu'il utilise pour Instagram. Le 27 octobre, Instagram a répondu « trop de requêtes ». Depuis début novembre, plus jamais Agent Reach sur Instagram avec ce compte : une restriction nous coûterait bien plus que ce qu'on y lit.
- Lemo Opuscar fait partie des outils de film en code qu'on a essayés. On a copié le dépôt le 14 octobre 2026, dans une version plus ancienne, sous licence MIT à l'époque, et fait plusieurs films avec ses styles : estampe, lanterne de papier, trait unique, maquette, gravure. On s'est aussi servi de sa banque de sons libres pour la musique de plusieurs vidéos.
- FreeLLMAPI, Ponytail et SkillSpector : pas lancés pour ce guide.

---

8. La méthode

Guide écrit le 14 novembre 2026, relu le 23 novembre avant sa mise en ligne (les cinq dépôts par l'API GitHub : licences, chiffres des README, offre Premium de FreeLLMAPI, commandes d'installation ; seule celle de SkillSpector avait changé). Aucun des cinq outils n'a été installé ni lancé, aucun dépôt cloné. Lus par l'API GitHub entre 13 h 58 et 14 h 10 : les chiffres des cinq dépôts, leurs dernières versions, leurs fichiers LICENSE, et pour chacun les documents cités (revue des conditions et commande npx de FreeLLMAPI, licence de Lemo Opuscar, banc d'essai

de Ponytail du 7 octobre, guide d'installation et doc des sessions d'Agent Reach, README de SkillSpector). Relus le même jour : l'article arXiv 2601.10338 en entier, les conditions de X, de Reddit, de LinkedIn et de l'API Gemini, la page de la doc de Claude Code sur les passerelles, la page de NVIDIA sur le scan des skills et la page des tarifs de Claude. Les pages de Cerebras ont été relues le 7 novembre.
