Jumeau numérique · Sous-section

Utiliser Codex dans son coffre

Alternative à OpenCode Desktop pour les utilisateurs ChatGPT : installer Codex (la boucle agentique d'OpenAI), le connecter à ton abonnement ChatGPT via login (pas de clé API), configurer les MCP et brancher les skills de ton coffre.

Source : https://prisme.one/plateforme/ia-juillet-2026/semaine-1-codex-coffre

Objectif

Codex ne lit les skills qu'à un seul endroit. Tu demandes à ton IA de créer un symlink vers ton vrai dossier de skills, et c'est réglé.

> **Cette leçon s'adresse aux utilisateurs de Codex.** OpenCode Desktop reste la stack recommandée par défaut dans ce bootcamp. Mais si tu as un abonnement ChatGPT Plus ou Pro et que tu veux la boucle agentique d'OpenAI dans ton terminal, Codex (l'agent CLI de ChatGPT) s'installe proprement à côté de ton coffre. Cette leçon te montre comment, en te reposant au maximum sur ton IA.

Utiliser Codex dans son coffre

Utiliser Codex avec ton système

LOOM

Utiliser Codex avec ton système

https://www.loom.com/share/d86c453a7cc744babac608030f79d34d

La méthode rapide : rien ne change, tu rebranches juste tes MCP et tes skills, et tu laisses ton IA faire le travail.

L'idée en une phrase

Bonne nouvelle : **presque rien ne change.** Le jour où tu passes d'OpenCode à Codex (ou à Claude Code, ou à un autre agent), tu ouvres l'agent dans ton dossier et tout ton contexte est déjà là : ton `AGENTS.md`, tes notes, tes projets. Tu n'as pas à tout reconfigurer.

Les **deux seules choses** à rebrancher quand tu changes d'agent, ce sont **tes MCP** et **tes skills**. Pourquoi ? Parce que chaque outil a sa propre convention pour ranger ces deux-là. Tout le reste marche pareil dès que tu ouvres le bon dossier.

À rebrancher #1
Tes MCP (les outils)

Tu demandes à ton IA de reprendre ta config MCP existante (Claude / OpenCode) et de la transvaser dans Codex. Tu ne touches à rien à la main.

À rebrancher #2
Tes skills (les commandes)

Codex ne lit les skills qu'à un seul endroit. Tu demandes à ton IA de créer un symlink vers ton vrai dossier de skills, et c'est réglé.

Le reste de cette leçon, c'est juste ça : te connecter avec ton abonnement, puis rebrancher ces deux choses en laissant l'IA faire le travail.

Étape 1 : Se connecter avec ton abonnement ChatGPT

C'est le gros intérêt de Codex : tu te connectes avec **ton compte ChatGPT** (login), pas avec une clé API facturée au token.

Au premier lancement, tape `codex`, puis choisis **Sign in with ChatGPT**. Ton navigateur s'ouvre, tu autorises, c'est réglé. Codex tourne alors dans le cadre de ton abonnement Plus / Pro / Business, **sans surcoût au token**.

> **Pourquoi ça compte :** l'autre méthode, c'est une clé API OpenAI facturée à chaque token consommé, et ça chiffre vite. Avec le login ChatGPT, tu réutilises l'abonnement que tu paies déjà. Pour le détail abonnement vs API, voir Structure des coûts →.

Étape 2 : Installer et lancer Codex dans ton coffre

Codex CLI s'installe en une commande :

```
# macOS / Linux (installeur officiel)
curl -fsSL https://chatgpt.com/codex/install.sh | sh

# ou via npm (multi-plateforme)
npm install -g @openai/codex

# ou via Homebrew (macOS)
brew install --cask codex
```

Sur Windows :

```
powershell -ExecutionPolicy ByPass -c "irm
https://chatgpt.com/codex/install.ps1 | iex"
```

Ensuite tu ouvres Codex **dans le dossier où tu veux travailler** (c'est le principe le plus important : l'IA voit ce qui est dans le dossier où tu la lances) :

```
# Va dans ton Workspace racine, puis lance Codex
cd ~/Documents/WORKSPACE && codex
```

Codex lit l'`AGENTS.md` à la racine (son fichier d'instructions natif) et a accès à tout ton coffre.

> **Astuce :** crée un alias `cox` dans ton `.zshrc` : `alias cox="cd ~/Documents/WORKSPACE && codex"`. Tu tapes `cox` de n'importe où, tu atterris dans ton coffre avec Codex lancé.

Étape 3 : Rebrancher tes MCP (laisse ton IA le faire)

Tu as déjà tes MCP configurés dans Claude Code ou OpenCode. Pas besoin de tout refaire à la main : **demande à ton IA de les transvaser dans Codex.** Colle-lui ce prompt :

```
J'utilise déjà des MCP dans Claude Code et/ou OpenCode. Reprends
ma
```

```
configuration MCP existante et transvase-la dans Codex, dans mon
fichier
~/.codex/config.toml au niveau user (global), pour que ces MCP
soient
actifs dans tous mes projets. Vérifie ensuite avec "codex mcp
list".
```

L'IA regarde ta config existante, la traduit au format de Codex, et c'est fonctionnel.

Où c'est rangé ? Codex stocke ses MCP dans un fichier `config.toml`, à deux niveaux possibles :

- **`~/.codex/config.toml`** (niveau ordinateur, **recommandé**) : tes MCP sont dispos dans toutes tes sessions, quel que soit le dossier.
- **`.codex/config.toml`** dans un projet : des MCP actifs **uniquement** pour ce projet (pratique pour séparer un MCP pro d'un MCP perso).

> L'app Codex a aussi un onglet **Plug-ins** où tu peux voir et ajouter des MCP à la main. Mais le plus rapide reste de laisser l'IA s'auto-configurer : elle sait le faire, autant ne pas perdre de temps avec ça.

Étape 4 : Rebrancher tes skills (un simple symlink)

Petit piège à comprendre. Dans OpenCode, tu peux ranger tes skills **où tu veux** : le fichier `opencode.json` déclare le chemin, et l'IA va les chercher là. C'est parfait, tu synchronises tes skills comme tu veux.

Codex, lui (comme Claude Code et ChatGPT), ne fonctionne pas comme ça : il ne regarde les skills qu'à **un seul endroit imposé**, `.agents/skills`. Si tes skills sont ailleurs (dans

`Second Cerveau/4 TOOLS/skills`, la convention du bootcamp), Codex ne les voit pas.

La solution, c'est la même que pour Claude Code : un **symlink** (un lien miroir) entre `.agents/skills` et ton vrai dossier de skills. Fais-le faire par ton IA, colle-lui ce prompt :

```
Crée un symlink pour que Codex voie mes skills. Mes skills sont
dans
"Second Cerveau/4 TOOLS/skills" à la racine de mon Workspace. Crée
un lien
symbolique ".agents/skills" qui pointe vers ce dossier (au niveau
du
Workspace, ou dans ~/.agents/skills si tu veux qu'ils soient
dispos dans
TOUS mes projets). Vérifie ensuite que le lien pointe au bon
endroit.
```

Une fois le symlink créé, Codex croit qu'il y a des skills rangés dans `.agents/skills`, alors qu'en réalité ils vivent dans ton coffre. Tu tapes `/skills` (ou `$` pour mentionner un skill par son nom) dans Codex, tu vois ta liste.

> **La philosophie du bootcamp :** évite de ranger tes skills **directement** dans `.agents/skills`. Sinon ils restent coincés dans Codex / ChatGPT. Mieux vaut qu'ils restent **transverses**, dans ton second cerveau, dispos pour n'importe quel agent via le symlink. Ton système reste souverain, pas prisonnier d'un outil.

Le faire à la main (Mac/Linux + Windows)

Récap

1.
Installer Codex
`npm install -g @openai/codex` (ou Homebrew / installeur)
2.
Se connecter avec ChatGPT
Sign in with ChatGPT, jamais une clé API
3.
Rebrancher les MCP
Demander à l'IA de transvaser ta config dans Codex
4.
Rebrancher les skills
Demander à l'IA le symlink `.agents/skills`
5.
Ouvrir Codex dans le Workspace
`cd ~/Documents/WORKSPACE && codex`

Retiens le principe : **quel que soit l'agent** vers lequel tu vas (Claude Code, Codex, un autre), les deux seules choses à rebrancher sont **les MCP et les skills**. Chacun a ses conventions pour ces deux-là. Le reste, tu ouvres dans le bon dossier et ça fonctionne.

- [ ] Je sais quand utiliser Codex (abonnement ChatGPT + boucle agentique GPT dans mon terminal)
- [ ] Codex CLI installé (npm install -g @openai/codex, Homebrew, ou l'installeur officiel)
- [ ] Je me connecte avec Sign in with ChatGPT (mon abonnement, pas une clé API facturée au token)
- [ ] Mes MCP rebranchés : config transvasée par l'IA dans ~/.codex/config.toml
- [ ] Mes skills rebranchés : symlink .agents/skills → Second Cerveau/4 TOOLS/skills
- [ ] Codex lancé depuis mon Workspace : je vois mes skills via /skills ou $

Sources

- Codex CLI (installation + connexion ChatGPT)
- Connecter Codex à des outils via MCP
- Créer et connecter des skills à Codex
- Configuration de base (config.toml)

- Dépôt open source Codex (openai/codex)

Vidéos

LOOM

Utiliser Codex avec ton système

https://www.loom.com/share/d86c453a7cc744babac608030f79d34d

Extraction verbatim depuis Prisme One · https://prisme.one/plateforme/ia-juillet-2026/semaine-1-codex-coffre