

接続マニュアル

Claude Code → Codex → Computer Use 接続マニュアル v1.0

APIキーを1つも使わずに、Claude Codeの中からCodex（GPT-6 Astra）を呼び、画像生成とパソコンの操作まで動かします。このマニュアルは、**Claude Codeは使っているがCodexは触ったことがない方**を想定して書いています。WindowsとMacの両方の手順を載せています。

使い方：各章の「Claude Codeに貼る文」を、そのままClaude Codeに貼り付けてください。あとはClaude Codeが進めます。手を動かすのは「ログイン」と「許可のボタンを押す」ところだけです。

もくじ

1. これは何をするものか
2. 用意するもの（Windows／Mac）
3. Codex CLIを入れる
4. Claude CodeからCodexを呼ぶ
5. モデルをGPT-6 Astraに切り替える
6. つながったか、1行で確かめる
7. 許可ファイルの書き方
8. --approve-for-me を付けてよい仕事の判断表
9. Computer Useを許可する（Chrome）
10. 詰まった6カ所の直し方
11. 終わったら閉じる
12. よくある質問

1. これは何をするものか

Claude Codeが直接触れるのは、テキストとファイルとフォルダまでです。絵は描けず、パソコンの中のアプリも触れません。その2つを持っているのがCodexで、Codexは**Claude Codeのターミナルの中から呼ばま**

す。

```
Claude Code (文章・コード・ファイル)
|
|   codex exec で仕事を1つ渡す
▼
Codex (GPT-6 Astra・画像)
|
|   Computer Use / Chromeプラグイン
▼
パソコンの操作 (メモ帳・Photoshop・Chrome ...)
```

手元のメインはClaude Codeのまま。足りないところだけCodexに取りに行きます。**お金の話**：CodexはChatGPTのアカウントでログインして、月額プランの枠で動きます。APIキーは要りません。

2. 用意するもの (Windows／Mac)

何	Windows	Mac
ChatGPTの有料プラン	Plus以上（私はPlusで動かしました）	同じ
Node.js	18以上（ <code>node -v</code> で確認）	同じ
Claude Code	入っていて、Bash（ターミナル）が使える状態	同じ
Codexデスクトップアプリ （ChatGPTアプリ）	Microsoft Storeから	公式サイトから
Chrome	拡張機能「ChatGPT」を入れる （Chromeウェブストア）	同じ

デスクトップアプリは「パソコンを動かす」ために必要です。Claude Codeから呼ぶのはCLI（ターミナルのcodexコマンド）ですが、Computer UseのプラグインはアプリにしかないためアプリもCLIも両方入れます。

3. Codex CLIを入れる

Claude Codeに貼る文

Codex CLI を入れたいです。次の順に進めてください。

1. ``npm i -g @openai/codex@latest`` を実行する
2. ``codex --version`` を実行して、0.154.0 以上であることを確かめる（それより古いと gpt-6-astra が使えません）
3. 私に「ターミナルで codex と打って、ChatGPT のアカウントでログインしてください」と案内して止まる

ログインはブラウザが開きます。APIキーではなく、ChatGPTのログインです。

Macの補足：`codex` が見つからないときは、npmのグローバルの場所がPATHに入っていない。 `npm bin -g`（または `npm prefix -g`）で出るフォルダを `~/.zshrc` の `export PATH` に足してください。

4. Claude CodeからCodexを呼ぶ

呼び方は2つあります。このマニュアルは②（直接叩く）で進めます。

- ① Claude Codeのプラグイン：`/plugin marketplace add openai/codex-plugin-cc` → codexをインストール。Windowsでは結果が空のまま返ることがあります（公開されているissueがあります）
- ② Claude CodeのBashから `codex` コマンドを直接叩く：OSを問わず動きます

基本の形

```
codex exec --skip-git-repo-check -m gpt-6-astra - < task.md
```

- `task.md` に頼みたいことを書いておき、`-` で標準入力から渡します
- `--skip-git-repo-check` は「gitのフォルダでなくても動かす」印です
- 結果は標準出力に返ります。Claude Codeはそれを読んで要約します

Claude Codeに貼る文

これから Codex を Bash から直接呼びます。プラグインは使いません。
頼みごとがあるときは、内容を `task.md` に書いてから

```
`codex exec --skip-git-repo-check -m gpt-6-astra - < task.md`
```

を実行し、返ってきた内容を要約して報告してください。

5. モデルをGPT-6 Astraに切り替える

毎回 `-m gpt-6-astra` を付ければ切り替わります。付け忘れると既定のモデルで動きます。

毎回付けるのが面倒なら、Codexの設定ファイルに書いておきます。

OS	設定ファイル
Windows	<code>%USERPROFILE%\codex\config.toml</code>
Mac	<code>~/codex/config.toml</code>

```
model = "gpt-6-astra"
```

この1行を足すと、`-m` を付けなくてもAstraで動きます。何のモデルで動いたかは、`codex exec` の冒頭に出る `model:` の行で確認できます。

6. つながったか、1行で確かめる

いきなり大きな仕事を渡さず、まず1行だけ渡してください。

Claude Codeに貼る文

```
次の1行を実行して、返ってきた文をそのまま見せてください。  
codex exec --skip-git-repo-check -m gpt-6-astra "今日の日付を日本語で1行"
```

私の環境では7秒で日付が1行返ってきました。これが返れば、Claude CodeとCodexはつながっています。返らないときは、10章の「詰まった6カ所」を先に見てください。つながっていないのに先へ進むと、あとで切り分けができなくなります。

7. 許可ファイルの書き方

Computer Useは、**アプリ1つずつ**許可を出す設計です。初めてのアプリを動かそうとすると「Allow Codex to use ○○?」と聞かれ、選択肢は4つです。

選択肢	意味
Allow	今回だけ
Allow for this session	このセッションだけ
Always allow	ずっと許可（設定ファイルに1行追加される）
Cancel	やめる

「Always allow」を選ぶと、次のファイルに1行ずつ溜まっていきます。

OS	許可ファイル
Windows	<code>%USERPROFILE%\codex\computer-use\config.toml</code>
Mac	<code>~/codex/computer-use/config.toml</code>

私のWindowsの環境では、こうなっています。

```
[apps]
allowed = ["notepad.exe", "Microsoft.WindowsNotepad_8wekyb3d8bbwe!App",
"chrome.exe"]
```

先に書いておけば、Claude Codeから呼んだときに許可の画面で止まりません。Claude Codeから呼んだCodexは対話画面を持たないので、ここで止まると自動で「拒否」になります。

- 同じメモ帳でも2回聞かれます。`notepad.exe` で起動したものと、ストア版のメモ帳は別のアプリとして扱われるからです。バグではなく「別の実体は別の許可」という設計です
- Macの書き方は、1回「Always allow」を押して、このファイルに書き込まれた行をそのまま写してください。アプリの名前の書き方がWindowsと違うため、手で推測して書かないほうが確実です
- 何を渡していて何を渡していないかは、このファイルを見れば分かります。**渡すアプリは最小にするのが**おすすめです（私はメモ帳とChromeだけ）

8. --approve-for-me を付けてよい仕事の判断表

Chromeプラグインは、初めてのサイトを開くときに「このサイトにアクセスしていいですか」と聞いてきます。Claude Codeから呼んだCodexはこれに答えられず、自動で「拒否」になります。

対策が `--approve-for-me` です。人の代わりに自動のレビューが許可・拒否を判断します（私の環境の0.154.0で使えました。公式リファレンスには載っていないので、`codex exec --help` で確認してから使ってください）。

```
codex exec --skip-git-repo-check --approve-for-me -m gpt-6-astra - < task.md
```

これは「人の判断を自動に置き換える」スイッチです。何を渡すかは、付ける前に自分で決めてください。私の線引きはこうです。

仕事	付けるか	理由
自分のChatGPTで画像を作る	付ける	失敗しても何も送信されない
開いているタブ・ファイルを読んで要約する	付ける	読むだけ。外に出ない
手元のファイルに下書きを書く	付ける	手元で見てから使える
メール・LINE・SNSに送信・投稿する	付けない	取り消せない
買う・申し込む・削除する	付けない	取り消せない
ログイン情報・パスワードを扱う	付けない	Computer Use自身も止める設計になっている
他人のアカウント・共有フォルダを触る	付けない	自分の範囲を越える

迷ったら付けない。付けずに止まったら、その仕事は人が見て進めるものだった、ということです。

9. Computer Useを許可する（Chrome）

パソコンを動かすには、Codexデスクトップアプリを1回起動してプラグインを入れます。CLIだけでは動きません。

1. Codexデスクトップアプリ（ChatGPTアプリ）を入れて、1回起動する
2. 左のメニューの「Plugins」から「**Computer Use**」を入れる
3. 同じ「Plugins」から「**Chrome**」を入れる（一覧の名前は「Browser」ではなく「Chrome」。
「Browser」はアプリ内蔵ブラウザ用で最初から有効）
4. Chromeに拡張機能「ChatGPT」を入れる（Chromeウェブストア）
5. OS側の画面操作の許可を出す
 - **Mac**：システム設定のポップアップで「画面収録」と「アクセシビリティ」を許可する
 - **Windows**：初めてアプリを動かすときの「許可の画面」（7章）で出ます

アプリを1回でも起動すると、Codex CLIの設定ファイル（`~/.codex/config.toml`）にComputer Use用の設定が自動で書き込まれます。CLIはこの設定を読むので、以後はClaude Codeから呼んだCodexでもComputer Useが使えます。アプリは起動したままでなくて構いません。

Windowsの人に2つ

- アプリ本体のフォルダ（`Program Files\WindowsApps` の下）にある `codex.exe` は直接実行できません（`Access is denied`）。アプリが外に展開している
`%LOCALAPPDATA%\OpenAI\Codex\bin\<英数字>\codex.exe` を使うか、npm版の `codex` をそのまま使ってください
- **Windowsでは、ブラウザだけComputer Useが自分で止まります**（「could not determine the current browser URL on Windows」）。Windowsでブラウザを動かすときは、Computer UseではなくChromeプラグイン（上の3と4）の担当です。デスクトップのアプリはComputer Use、ブラウザの中はChromeプラグイン、と覚えてください

最初の1回は、いちばん軽いアプリで

Claude Codeに貼る文

```
task.md に次の内容を書いて、
`codex exec --skip-git-repo-check -m gpt-6-astra - < task.md`
を実行し、返ってきた内容を要約してください。

--- task.md ---
Computer Use を使って、メモ帳（Windows は notepad、Mac は TextEdit）を起動し、
「接続テスト」と1行だけ打ってください。保存も閉じもしないでください。
終わったら、何をしたかを日本語で短く報告してください。
---
```

私の環境では「ウィンドウ一覧を取る → メモ帳を前面に出す → 入力 → 確認」の5手、82秒で終わりました。

10. 詰まった6カ所の直し方

6カ所とも「エラーが出ないか、出ても原因と関係ない文が出る」タイプです。

1. Claude Codeのプラグイン経由が、Windowsでは黙って空を返す

`codex-plugin-cc` を入れて呼ぶと、Windowsでは結果が空のまま返ることがあります。**直し方**：プラグインを使わず、Bashから `codex` コマンドを直接叩く（4章の形）。このマニュアルは全部そちらです。

2. 長い指示を引数で渡すと、8,191文字で無言で死ぬ

Windowsの `codex` はnpmの `codex.cmd` 経由で起動されるため、コマンド全体が8,191文字を越えると0秒・エラーなしで終了します。**直し方**：指示はファイルに書いて `-` で標準入力から渡す（`- < task.md`）。

3. 日本語を標準入力で渡すと、文字化けする

PowerShellから流すと日本語が化けます。**直し方**：流す前にPowerShellの出力をUTF-8にする。

```
[Console]::OutputEncoding = [Text.Encoding]::UTF8
$OutputEncoding = [Text.Encoding]::UTF8
```

Claude CodeのBashツールから流す分には、私の環境では化けませんでした。

4. `codex.exe` の場所が、アプリの更新で変わる

`%LOCALAPPDATA%\OpenAI\Codex\bin\<英数字>\codex.exe` の英数字は、アプリが更新されると変わります。**直し方**：固定パスで書かない。npm版の `codex` をそのまま使う。

5. 対話画面をプロジェクトのフォルダの中で開くと、先に2つ聞かれる

`codex` を対話モードで開くと「このフォルダを信頼するか」「フックを信頼するか」を先に聞かれます。直し方：試すときはプロジェクトの外の空のフォルダで開く。

6. 「Always allow」を選んでも、別に聞かれることがある

Computer Useの許可（アプリ単位）と、Chromeプラグインの許可（サイト単位）は別物です。直し方：Claude Codeから呼ぶ前に、Computer Useは許可ファイル（7章）へ、Chromeプラグインは `--approve-for-me`（8章）で、それぞれ手当てしておく。

おまけ：Windowsでファイルを書かせる仕事

私の環境では、Windowsの `codex` は `--sandbox workspace-write` を指定してもファイルを書けませんでした（読み取り専用に落ちる。上流に未解決のissueがあります）。ファイルを書かせたいときだけ `--sandbox danger-full-access` を付けています。付けるときは、作業フォルダを決めて、それ以外を触らせない指示を `task.md` に書いてください。

11. 終わったら閉じる

- Codexデスクトップアプリは閉じていて構いません（CLIは設定ファイルを読むだけ）
- 「Always allow」にしたアプリを取り消したいときは、許可ファイル（7章）の行を消すか、デスクトップアプリの設定のComputer Useの節から外します
- Chromeプラグインのサイト許可は、拡張機能「ChatGPT」の設定の許可・拒否リストから外せます

12. よくある質問

Q. APIキーは要りますか？ 要りません。CodexはChatGPTのアカウントでログインして、月額プランの枠で動きます。

Q. Plusでも動きますか？ 私はPlusで動かしました。公式の案内は「Work と Codex」と書いてあり、Plusの明記はありません。使えるかどうかは、6章の1行で確かめてください。

Q. どれくらい枠を使いますか？ 私の実測で、接続確認の1行は約4,000トークン、Chromeを動かしてChatGPTで画像を1枚作る仕事は約8万トークン・約2分半でした。失敗した回も同じように枠を使いま

す。前提（プラグイン・拡張機能・許可ファイル・`--approve-for-me`）を先に揃えてから走らせるほうが、結局いちばん節約になります。

Q. Macでも同じですか？ CLIの入れ方・呼び方・`-m gpt-6-astra`・`--approve-for-me` は同じです。違うのは、OS側の許可（画面収録・アクセシビリティ）と、許可ファイルのアプリ名の書き方です（7章）。Windowsで起きる「ブラウザだけComputer Useが止まる」は、Macでは起きませんでした（Macではアドレスバーからいま開いているURLを読めるため）。

Q. スキルにしていいますか？ 3段で確認してからにしてください。①Claude CodeからCodexがつながっている（6章）②CodexからComputer Useが使える（9章の最初の1回）③その上でスキルにする。1段飛ばすと、動かなかったときに「つながり方が悪いのか、やらせ方が悪いのか」が分かりません。

株式会社スターストリーム・スタジオ このマニュアルは公式LINEでお渡ししている方向けのものです。