Claude Code × Codex 連携ガイド

確認日: 2026年11月時点(公式のREADMEで確認)

Claude Codeの中から、OpenAIの「Codex」に見直しや作業を頼めるようにする手順です。

Claudeが書いたものを、別のAIの目でもう一度チェックできます。





先に早見表を置いておきます。スクショして手元に置いておくと便利です。

Claude × Codex 早見表

スクショして手元に置いておくと便利です

入れる(1回だけ)

```
/plugin marketplace add openai/codex-plugin-cc
/plugin install codex@openai-codex
/reload-plugins
/codex:setup
```

ログイン(初めてだけ)

```
!codex login
```

見直す・任せる

```
/codex:review
/codex:adversarial-review 心配なこと
/codex:rescue 頼みたいこと
```

時間がかかるとき

```
/codex:review --background
/codex:status
/codex:result
/codex:cancel
```

2026年11月時点・公式README(openai/codex-plugin-cc)で確認

準備するもの

- Claude Code(使える状態のもの)
- Codexを使えるChatGPTのアカウントか、OpenAIのAPIキー
- Node.js 18.18以上

※ ChatGPTでログインした場合は、プランのCodexの利用枠を使います。APIキーの場合は、API料金が別にかかります。

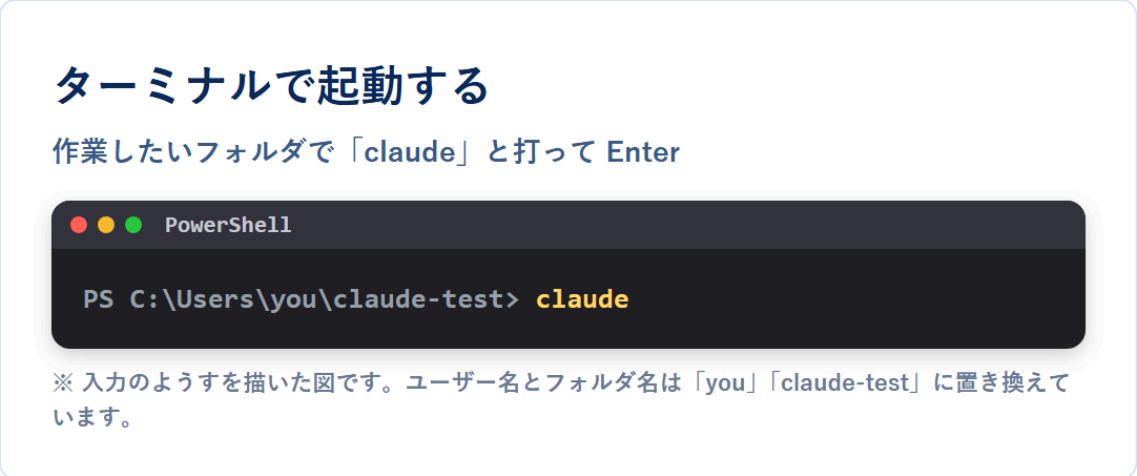
このガイドは、ターミナル(黒い画面)で動かすClaude Codeで進めます。

アプリ(デスクトップ版のCodeタブ)にも、プラグインを入れる画面があります。

このガイドでは、コピペで進められるターミナル版の手順を説明します。

ターミナルを開いて、これを送ると起動します。

```
claude
```



1. 入れる（4つを順番に）

起動したClaude Codeの入力欄に、1つずつ貼って送ります。

① プラグインの置き場所を登録

```
/plugin marketplace add openai/codex-plugin-cc
```

② プラグインを入れる

```
/plugin install codex@openai-codex
```

③ 読み込み直す

```
/reload-plugins
```

④ 準備できているか確認

```
/codex:setup
```

入力欄に1行ずつ送る

上から順番に。1つ送ったら、終わるのを待ってから次へ



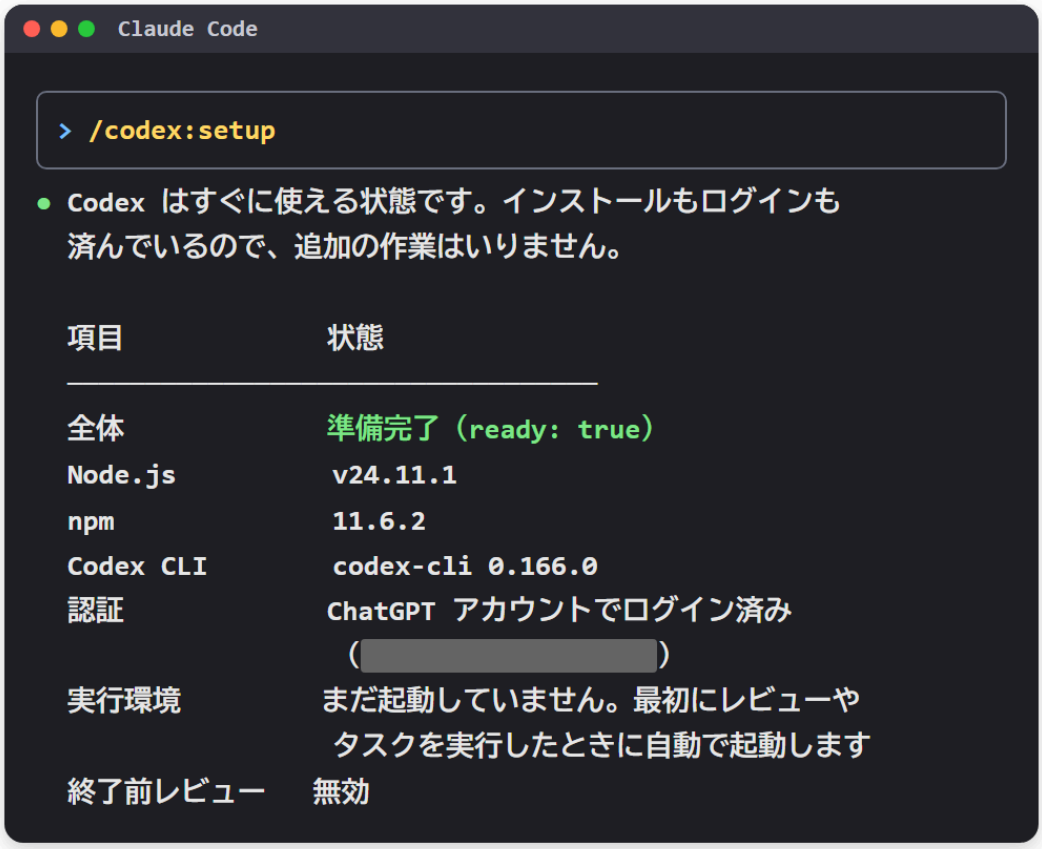
※ 入力欄のようすを描いた図です(返ってくる文は環境によって変わるので省いています)。

`/codex:setup` は、Codexが使える状態かを教えてくれます。

Codexが入っていない場合は、その場で入れるか聞いてくれます(npmが使える場合)。

④ /codex:setup の結果

「準備完了」と出ればOK。ログインがまだなら、ここで案内が出る



※ 練習用の空フォルダで実際に動かした出力を、読みやすい大きさで描き直しています。メールアドレスは伏せ、フォルダの場所は短くしています。

自分で入れたい人は、ターミナルでこれを送ります。

```
npm install -g @openai/codex
```

うまく入ると、`/codex:` で始まるコマンドが使えるようになります。

2. ログインする（初めての人だけ）

Codexにまだログインしていない場合は、これを送ります。

```
!codex login
```

ふつうはChatGPTのアカウントでログインします。

APIキーで使う人は、公式の認証の説明 (https://developers.openai.com/codex/auth)の手順で進めてください。

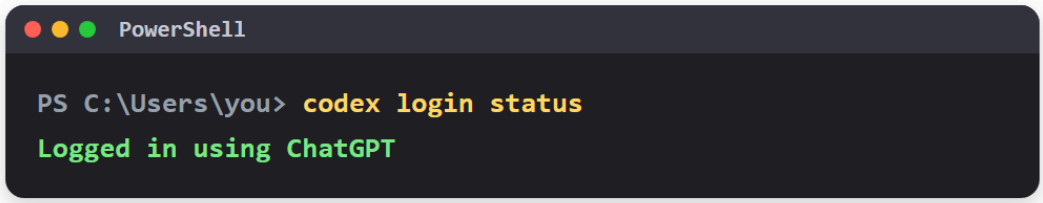
すでにこのパソコンでCodexにログインしている人は、そのまま使えます。

ログインできたかは、ターミナルでこれを送ると分かります。

```
codex login status
```

ログインできたか確かめる

ターミナルで「codex login status」。この表示ならログイン済み



※ 練習用の空フォルダで実際に動かした出力を、読みやすい大きさで描き直しています。メールアドレスは伏せ、フォルダの場所は短くしています。

3. 入ったか確かめる

試しに、いまの作業を見直してもらいます。

`/codex:review` が見るのは、Gitで管理しているフォルダの「まだコミットしていない変更」です。

はじめは、変更を1か所だけ入れた練習用のフォルダで試すと安心です。

```
/codex:review --background
```

進み具合を見る

```
/codex:status
```

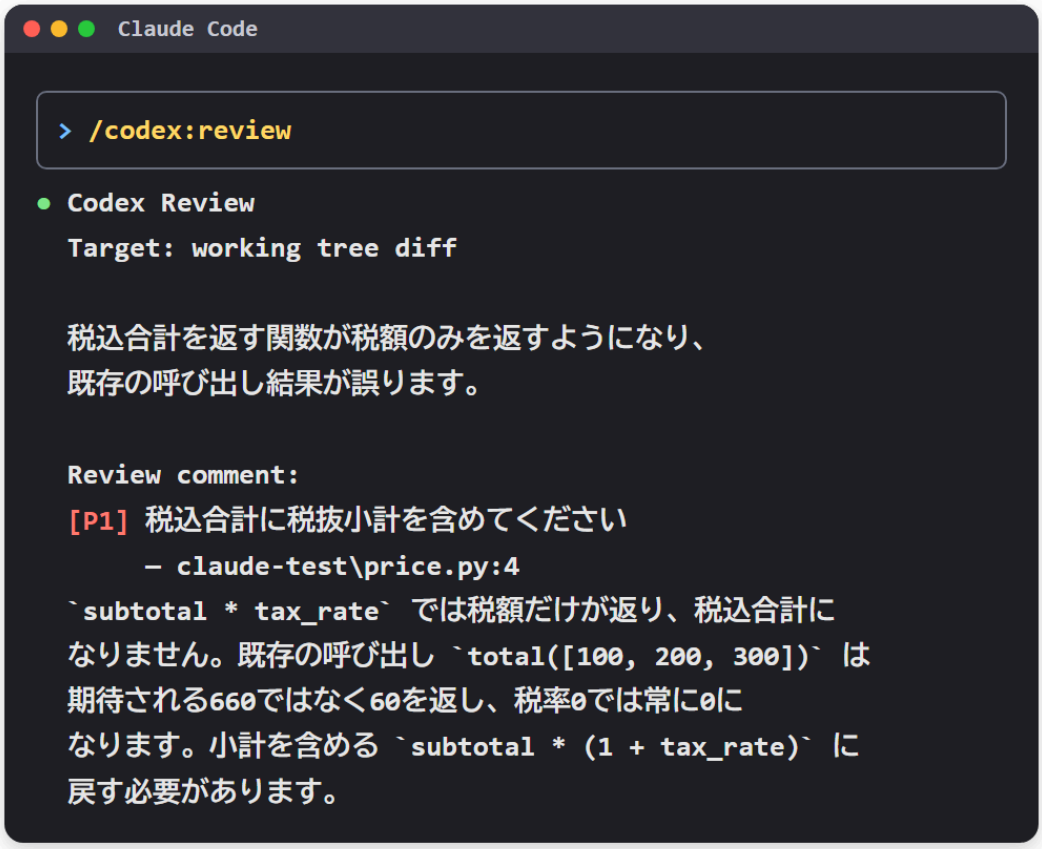
結果を見る

```
/codex:result
```

見直しが終わると、こんなふうに指摘が返ってきます。

/codex:review の結果

わざと計算ミスを入れた練習ファイルを見直してもらった例



※ 練習用のファイルで実際に動かした出力です。フォルダの場所は短くしています。[P1] は「優先度が高い指摘」の印です。

4. よく使う3つ

よく使う3つの使い分け

迷ったら、まず /codex:review

こんなとき	使うコマンド	書き換え
まだコミットしていない変更を見てほしい	`/codex:review`	しない
「ここが心配」と決めて厳しく見てほしい	`/codex:adversarial-review` ＋ 心配なことを書く	しない
調べもの・直す作業を任せたい	`/codex:rescue` ＋ 頼みたいことを書く	することがある

コードの見直し(書き換えない)

```
/codex:review
```

まだコミットしていない変更を見てくれます。

ブランチどうしを比べたいときは `/codex:review --base main` のように比べる先を付けます。

注文(「ここを見て」)は付けられません。

ねらいを決めて厳しく見直す(書き換えない)

```
/codex:adversarial-review
```

後ろに「何を疑ってほしいか」を書き足せます。

作業を任せる

```
/codex:rescue
```

後ろに頼みたいことを書きます。

「Codexに〜させて」とふつうの言葉で頼んでもOKです。

5. 文章を見直してもらうときの頼み方

`/codex:review` はコードの変更を見るためのものです。

台本や記事のような文章は、作業として頼みます。

たとえば、こう頼みます。

```
Codexに、この台本を事実確認の目でチェックさせて。
言い過ぎや誤解を招く所と、根拠と、直した文面の案を出して。
ファイルは書き換えないで。
```

Codexは指摘するだけにしてもらい、

採用するかを決めて直すのは、自分(とClaude)です。

6. つまずいたとき

- **動かない** → Node.jsが18.18以上か確かめる
- **`/codex:setup` だけでは進まない** → 2章の `!codex login` を送る
- **`/codex:` のコマンドが出てこない** → Claude Codeをいったん閉じて、もう一度 `claude` で起動する
- **時間がかかる** → 後ろに `--background` を付けて、`/codex:status` で様子を見る
- **途中で止めたい** →

```
/codex:cancel
```

7. 気をつけること

- 自動で何度も見直す設定(review gate)は、上限をすぐ使い切ることがあります。慣れるまでは使わないのがおすすめです。
- APIキーやパスワードは、チャットに貼らないでください。

つまずいた所があれば、LINEのトークにひとこと送ってください。

「どこで止まったか」だけでも大丈夫です！

出典

- openai/codex-plugin-cc(公式リポジトリ) (https://github.com/openai/codex-plugin-cc)
- Codexの料金とプラン(OpenAI公式) (https://developers.openai.com/codex/pricing)
- Codexの認証(OpenAI公式) (https://developers.openai.com/codex/auth)
- Claude Code デスクトップ版(Anthropic公式) (https://code.claude.com/docs/en/desktop)