



陸に進出したまんまんのらっこちゃん



ぼく、らっこちゃん！



かわいいだけじゃ満足できない向上心の獣！



いっしょにお勉強します！！

はじめに

◆ 作者紹介

貝 連太郎

ふだんは Maya を使って3Dでいろいろ作ってます。

大量のデジタル雑務に心がぐんにゃりしていたときにcursorに出会い、

自動化ツールをいろいろ作ってもらってめちゃくちゃ助けられています！

本書がcursorに触るきっかけになり、私と同じように困っている人の助けになればうれしいです。

cursorを紹介したお友達の感想も紹介します。

- ・ cursorがきてからコードほぼ書いていない
- ・ 簡単なプリント文もcursorに頼んでいる
- ・ ファイル名や変数名の名付けもcursorがやったほうがいい
- ・ cursorに課金しないと作業に支障がある
- ・ cursorが今まで知らなかつた機能を提案してくれる
- ・ 知らない言語でもcursorがいればたぶん大丈夫

わりと絶賛！



◆ この本の対象読者

以上のことから、本書は次のような方を対象としています。

- ・スクリプトを書けるようになりたいデザイナーの方
- ・業務の自動化に関心がある方
- ・プログラミング関連の書籍に挫折経験のある方
- ・スクリプトは多少書けるけどもっと楽に書きたい方

この本はcursorの使い方を通して、初心者の方でも楽しみながらプログラムを学べる本を目指して制作しました。より深くプログラミングを学びたい方は、別途、プログラミングの基礎を扱う専門書をご覧いただくことをおすすめします。

本書はWindows環境で作成していますが、Macでもほとんどの操作は同様に行えるはずです。

目次

はじめに	3
作者紹介	3
この本の対象読者	4
目次	6
一章 cursorをインストール！	8
cursorとは	9
インストールしよう	10
cursorを立ち上げてみよう	14
二章 最初のチャレンジ！	18
まずはフォルダーを開こう	19
時計アプリを作ってみよう	20
プログラムを保存・実行しよう	22
もしもうまくいかなかったら	26
プログラムをカスタマイズしてみよう	27
エラーに対応してみよう	30
ツールをデスクトップから起動しよう	33
これだけは覚えておこう！	35

三章 べんりなツールを作ろう	36
New Chat から始めよう	37
ファイルリストツールを作ろう	38
見た目をオシャレにしよう	41
エクセルチェックツールを作ろう	46
こんなツールも作れます	49
四章 知識を吸収させてみよう	50
最初に	51
Marpを使ってプレゼン資料を作ろう	52
知識の吸収方法いろいろ	60
過度の期待は禁物	62

一章 cursorをインストール！

cursorをインストールして下準備をしよう！

めんどくさい！泣いちゃうかも！



cursorとは

cursorとは: チャットで対話することでプログラミングしてくれ
る、次世代のエディタソフトです

主な特徴:

- ・書いてほしいスクリプトを勉強してくれる
- ・自動修正もしてくれる
- ・サジェストもしてくれる



なんだヒ――――!!!!

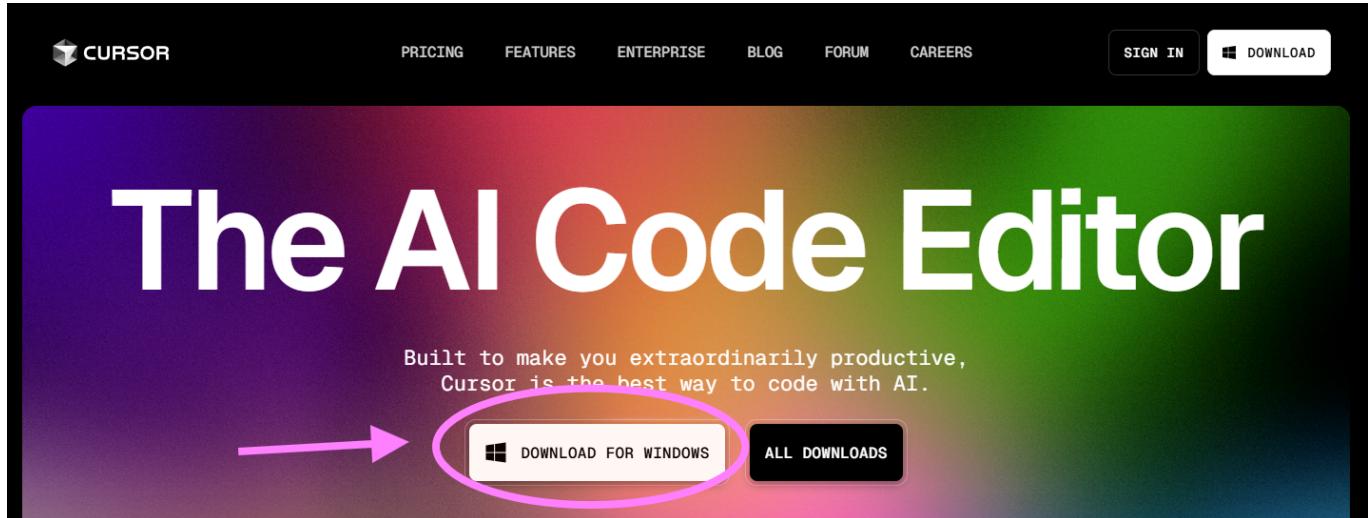
ネットから無料でダウンロードができ、最初の2週間は試用期間として有料機能を利用で
きます(2024年現在)。早速使っていきましょう！

インストールしよう

◆ cursorをインストールする

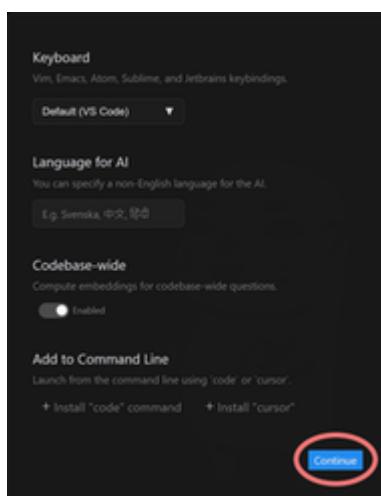
まず、cursorの公式サイトにアクセスしてください。

cursor公式サイト・<https://www.cursor.com/>

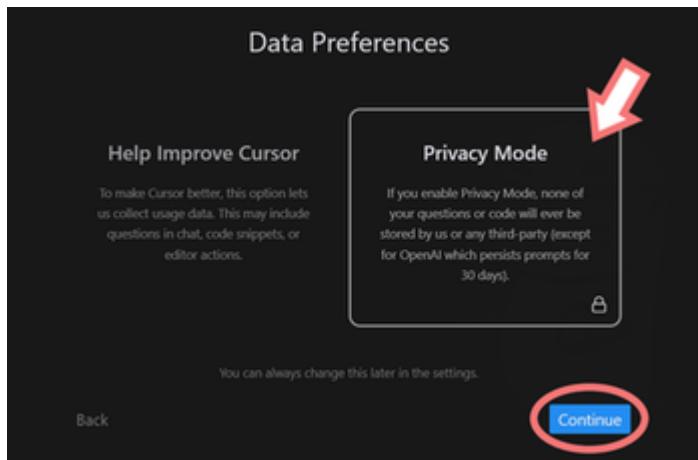


Downloadのアイコンを押してexeをダウンロードし、実行します。

最初の画面では、デフォルト設定のままcontinueを押しましょう。



DataPreference設定画面では プライバシーモードを選択しましょう。



これでcursorのインストールは完了です。

次はPythonをインストールしましょう。

♦ Pythonをインストールする



Pythonってなに！？
cursorだけじゃだめなの？

pythonは絶対絶対！必要なので、がんばってインストールしてください！



あ、そっちのばいそんね。うん！
(とりあえずわかったふりしちゃう)

一章 cursorをインストール！

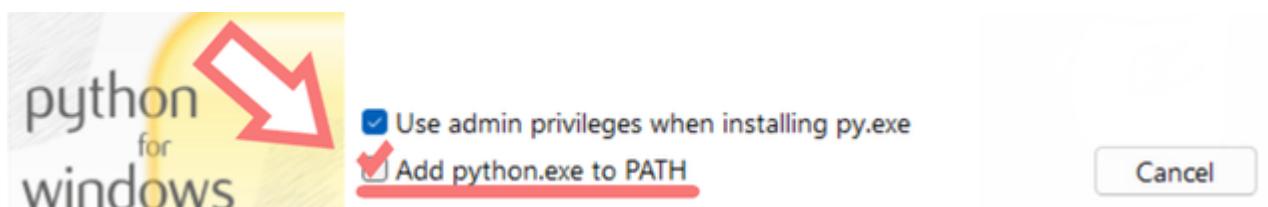
Python公式サイト・<https://www.python.org/>

The screenshot shows the Python.org homepage. At the top, there's a navigation bar with links for About, Downloads, Documentation, Community, Success Stories, News, and Events. Below the navigation bar, there's a code snippet example for generating a Fibonacci series up to n. To the right of the code, there's a section titled "Functions Defined" with a brief description of Python functions. Below the code example, there's a navigation bar with links 1 through 5. Underneath the code example, there's a large call-to-action button with the text "Python is a programming language that lets you work quickly and integrate systems more effectively. [» Learn More](#)". Below this, there are three main sections: "Get Started", "Download", and "Jobs". The "Download" section is highlighted with a red box around the "Latest: Python 3.12.5" link. A large red arrow points from this link towards the "Recommended" link in the "Files" section of the table below.

- ① 「Download」の「Latest」のリンクをクリック
- ②下の方にスクロールし、"Files" 項目に「Recommended」と書いてあるもののリンクを押してダウンロードし、実行。

Version	Operating System	Description	MD5 Sum	File Size
Gzipped source tarball	Source release		d23d56b51d36a9d51b2b13d30c849d00	25.7 MB
XZ compressed source tarball	Source release		02c7d269e077f4034963bba6befdc715	19.5 MB
macOS 64-bit universal2 installer	macOS	for macOS 10.9 and later	78bd8d0795062b1df63e2b8d8386a5fa	43.5 MB
<u>Windows installer (64-bit)</u>	Windows	<u>Recommended</u>	bccb2fcf9d739f776fb6414afc12c80d	25.3 MB
Windows installer (32-bit)	Windows		d151f5f116e11c4d40021527f51ddf67	24.0 MB

インストールは設定はおおむねそのままで良いですが、気をつけるのは、Add **python.exe to PATH** チェックボックスをONにすること。



お疲れ様です！

これでインストールは完了です。

次のセクションでは、cursorの基本的な使い方を紹介します。



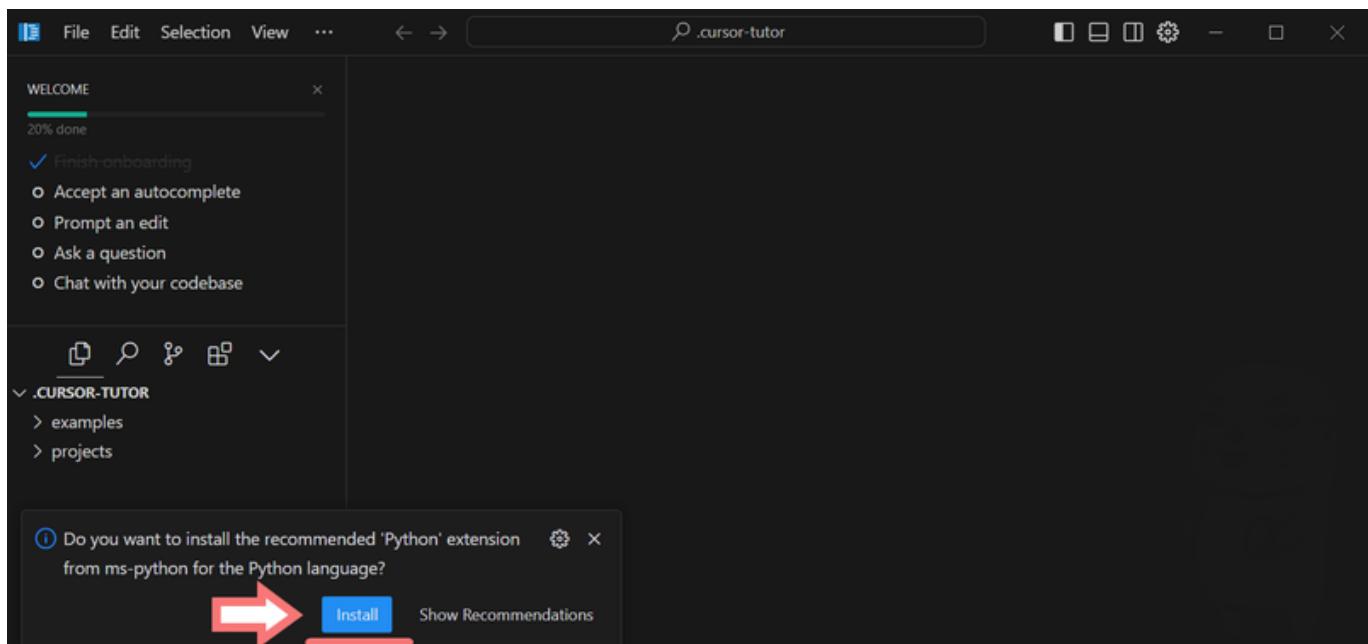
めちゃくちゃ頑張ったよ…！

いっぱい褒めてほしい！

cursorを立ち上げてみよう

◆ 画面を表示

cursorを起動します。こんな感じの画面が表示されると思います



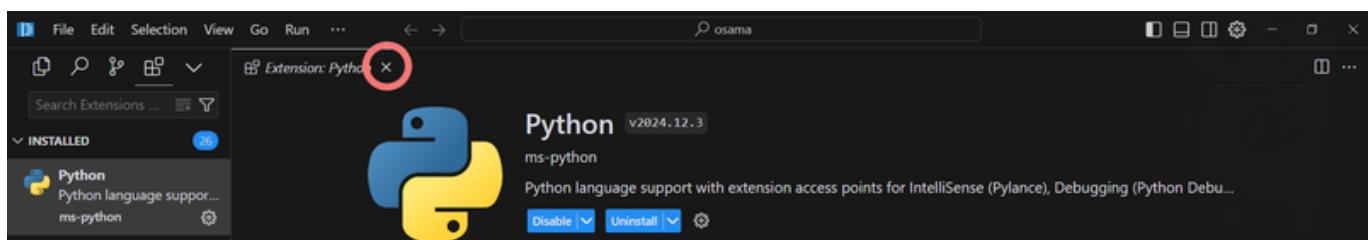
くろい…こわい…

謎の何かをインストールするよう促されている…！

「Python」は、ここでも必要なので「Install」を押しましょう！



うわ！また黒い画面でた…こわい！



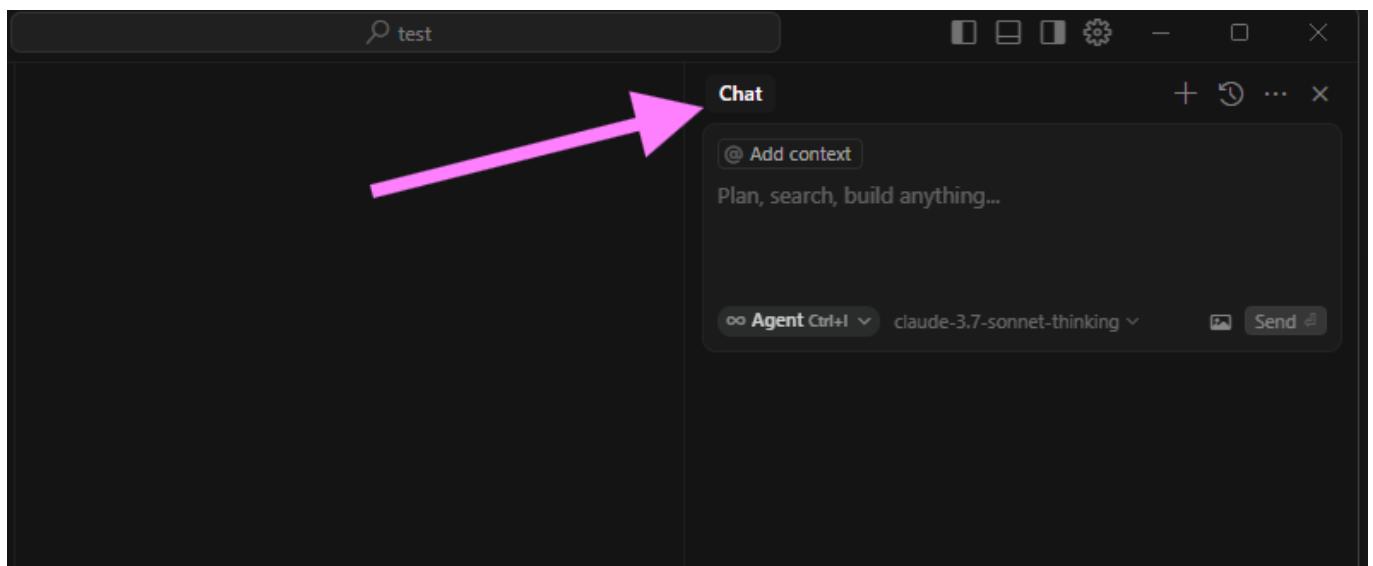
インストールするとこのような画面が出ます。今は×を押して閉じてください。

◆ チャット欄を出す

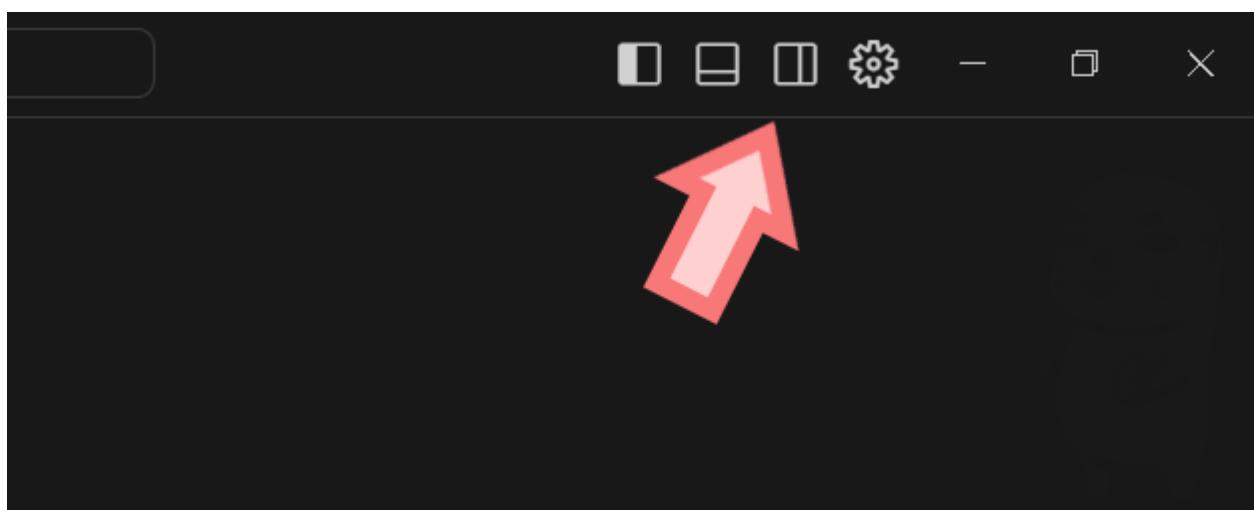
cursorはチャットでやりとりすることでプログラムを書いてくれます。

まずはチャットをする準備をはじめましょう。

画面の右側に Chatというウィンドウが出ていれば準備はOKです！



もしもこのウィンドウが出ていない場合は **右上の□アイコン** をクリックしcursorのチャット欄を表示してください。

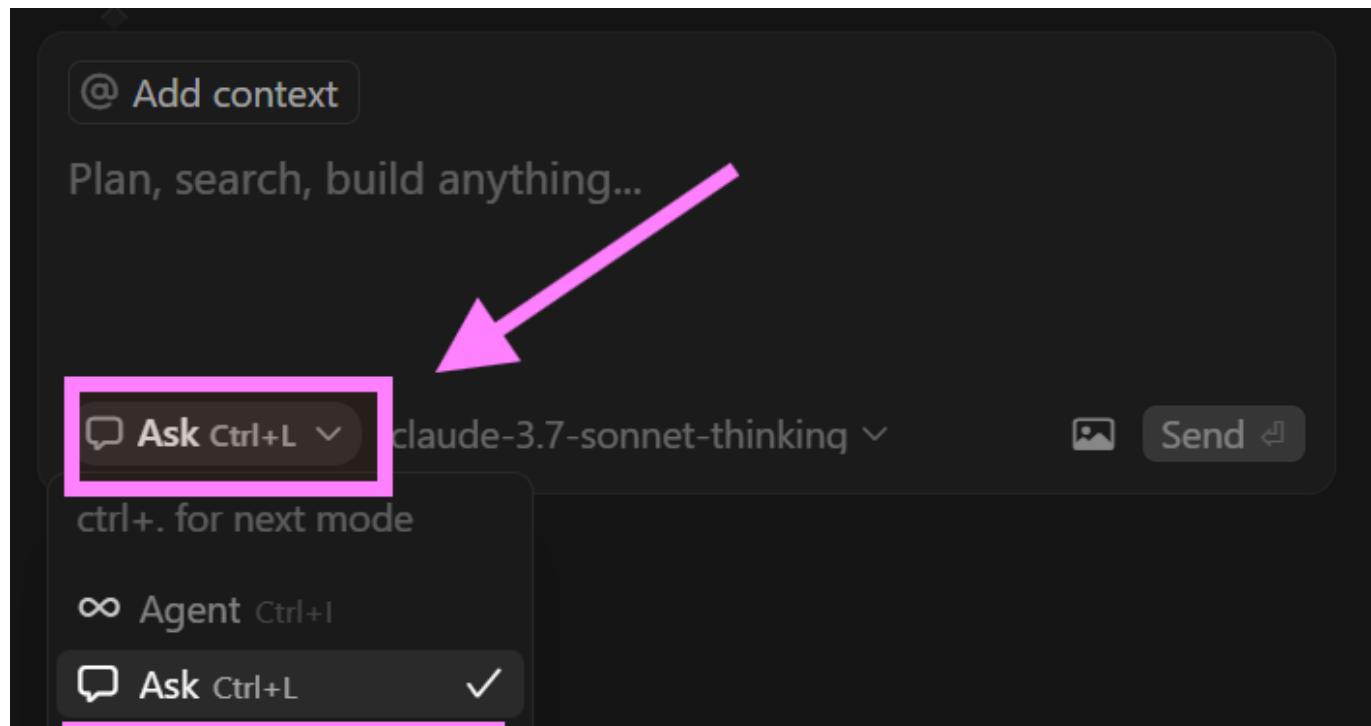


一章 cursorをインストール！

次に、チャット欄の左下が、最初は「Agent」になっているとおもいます。

Ask(旧称CHAT)に変更してください。

しばらくは、このAskを使ってチャットをしていきます。

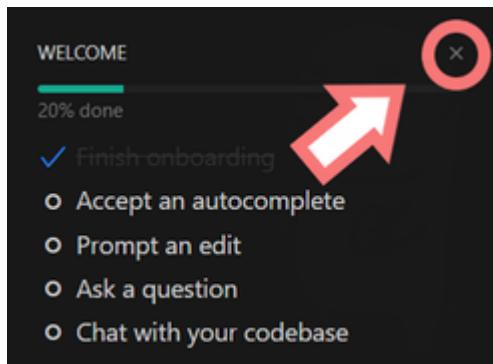


わーい！

これからいっぱいチャットしちゃうよ！

♦ welcomeは消してOK

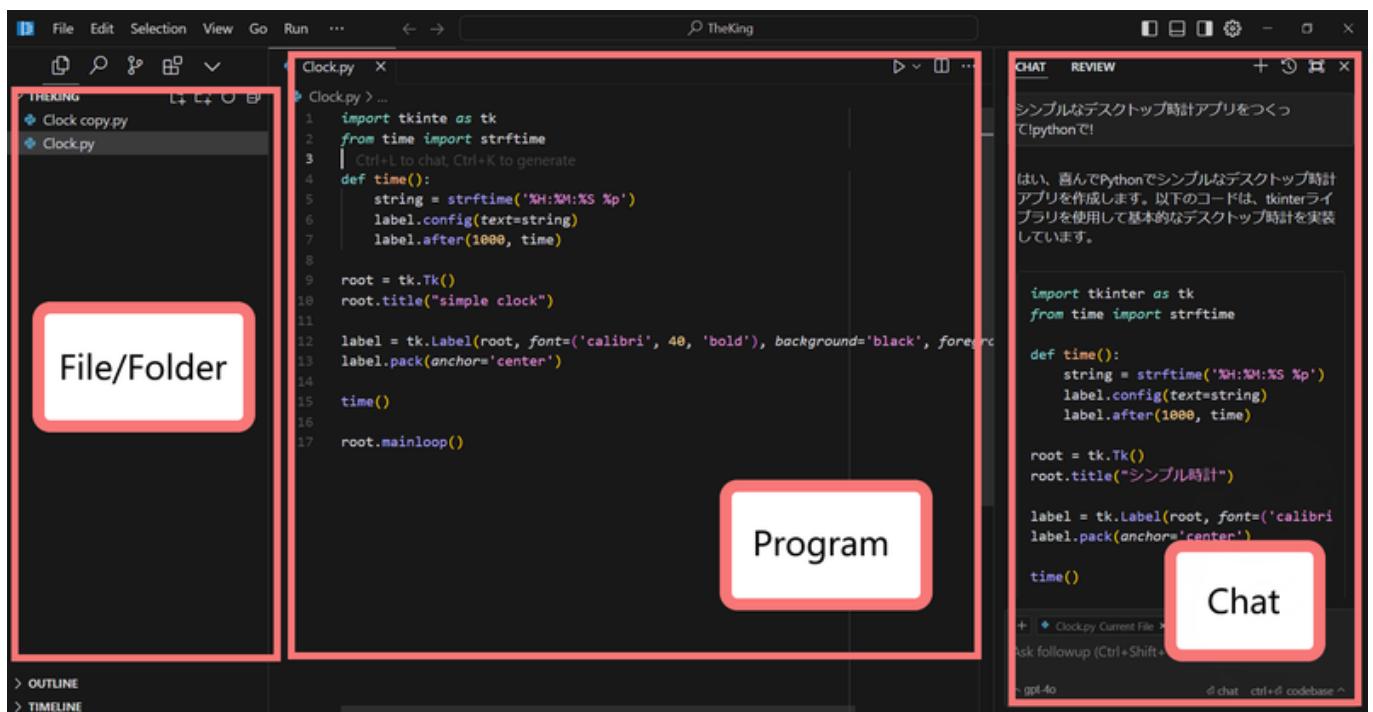
welcomeの画面が出ていたら、消してしまって大丈夫です。



♦ 画面の説明

画面は大きく分けて以下の3つの領域に分かれています。

1. 左側パネル：ファイルやフォルダの一覧を表示します。
2. 中央エディタ：ここでプログラムを作成し、修正します。
3. 右側チャット欄：Cursorのチャットインターフェースです。



二章 最初のチャレンジ！

cursorにプログラムを書いてもらおう！

完成するもの



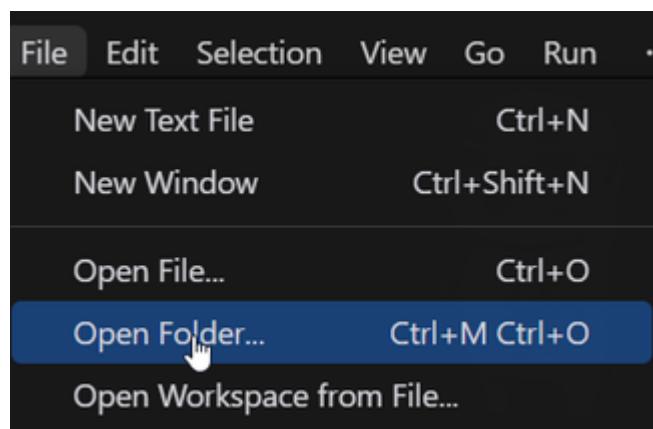
べつに時計はいらんが…？



まずはフォルダーを開こう

cursorはフォルダー単位で作業を行います。

File -> Open Folder を選択して、フォルダを開きましょう。



使うフォルダは、作業を始める前に「ClockFolder」のような名前で、空っぽのフォルダを適当な場所に作成しておいて、それを指定しましょう。

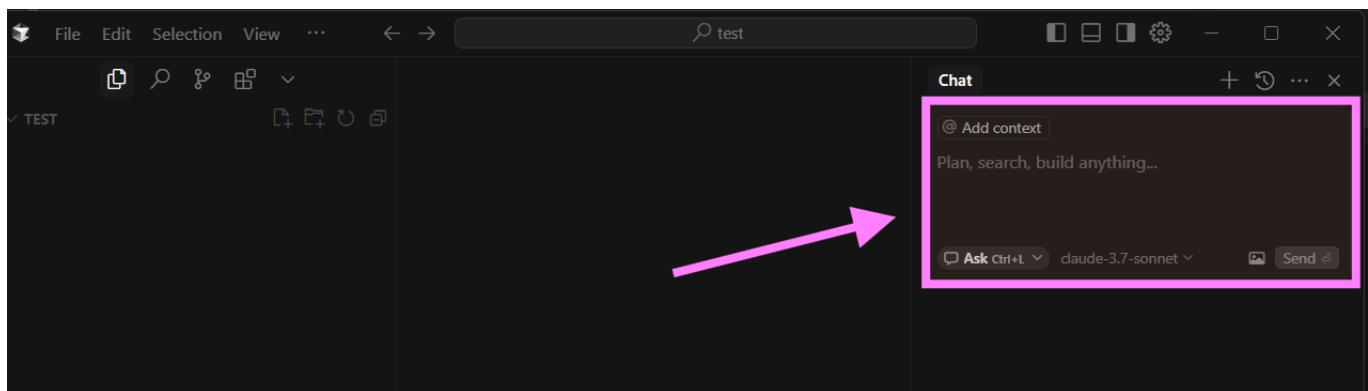
このフォルダに、すばらしい時計を作っていきますよ！

時計アプリを作つてみよう

♦ お願いを書く

時計を作ってくれというお願いを書いていきます。

まずは、**チャット欄を選択します。**



そして、下記のようにお願いを書いてください。

頼み方は、完全に同じ言い方で頼まなくとも大丈夫、ニュアンスがあっていればOKです。

@ Add context

シンプルなデスクトップ時計アプリをつくって!pythonで!

Ask claude-3.7-sonnet v

Send ↴

♦ cursorがコードを書く

お願いを書いたら**エンターを押す**と、cursorがコードを書きはじめます！

はい、喜んでPythonでシンプルなデスクトップ時計アプリを作成します。以下のコードは、tkinterライブラリを使用して基本的なデスクトップ時計を実装しています。

```
import tkinter as tk
from time import strftime

def time():
    string = strftime('%H:%M:%S %p')
    label.config(text=string)
    label.after(1000, time)

root = tk.Tk()
root.title("シンプル時計")

label = tk.Label(root, font=('calibri', 40, 'bold'),
background='black', foreground='white')
label.pack(anchor='center')

time()

root.mainloop()
```



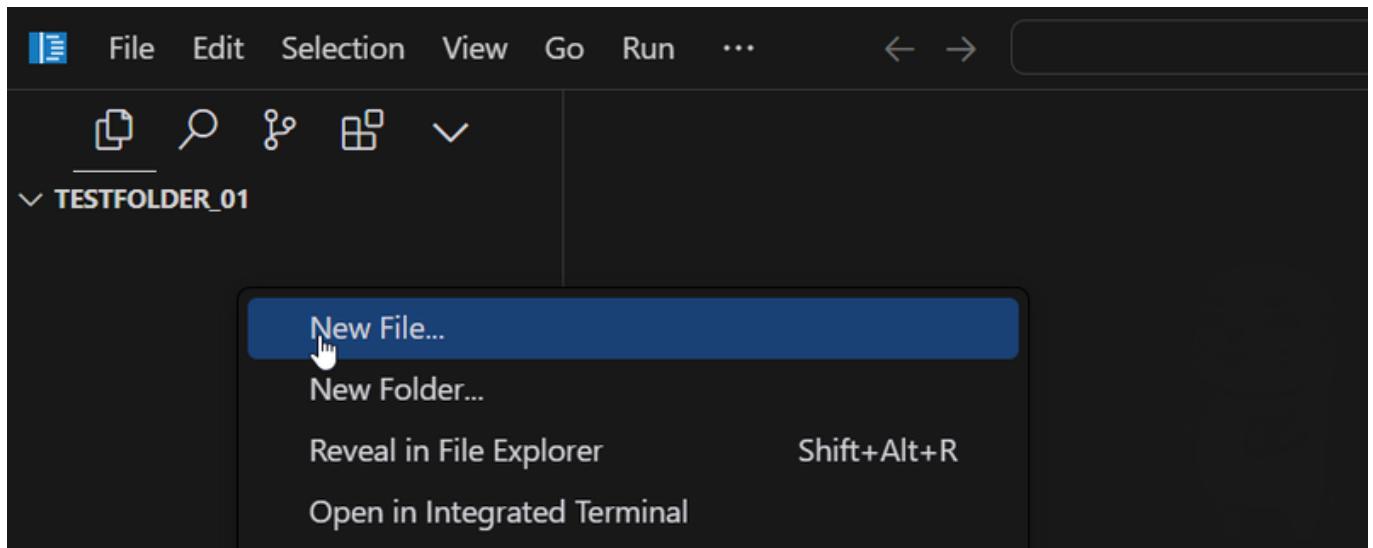
ふおおおおおおお！なんかずらすら書いてる！！！

プログラムを保存・実行しよう

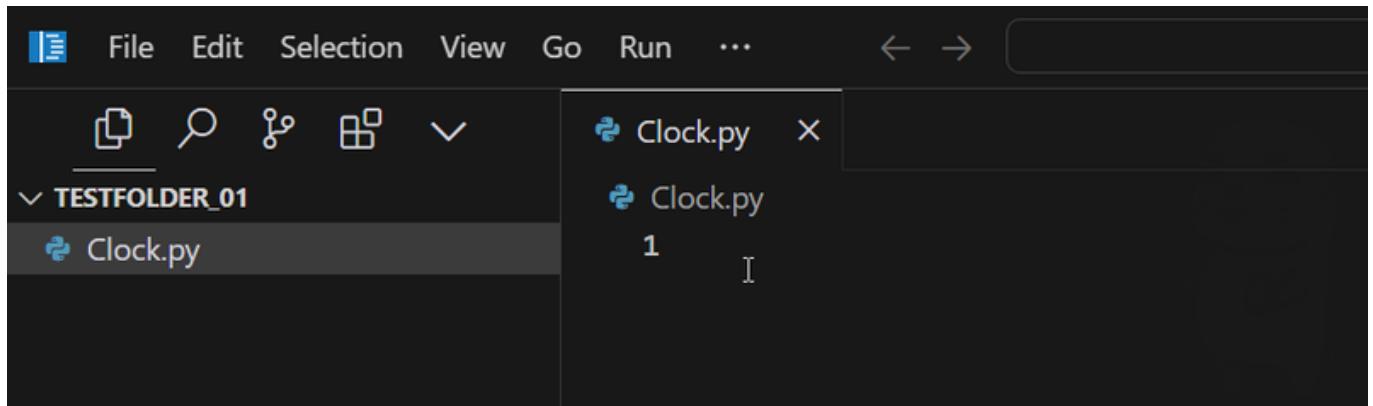
cursorがプログラムを書き終わったら、これを一回ファイルとして保存しましょう！ 空っぽのファイルを作って、そこにプログラムをコピペで貼り付けますよ！

◆ 空のファイルを作る

左側のパネル(ファイル/フォルダ画面)の上で、**右クリックする -> New File**



ファイル名を **Clock.py** とします。





.pyって拡張子だよね？

自分で書かなきゃダメなのかしら？

.pyはPythonのスクリプトファイルを示す拡張子です。

拡張子も自分で書きます！頑張って！

♦ プログラムを保存

Clock.pyに、今できたプログラムをコピーして貼り付けましょう。

chat欄の中の、プログラムが書いてある欄に、**カーソルを近づけていくと Copy のボタンが出てきます**。これを押してプログラムをコピーします。

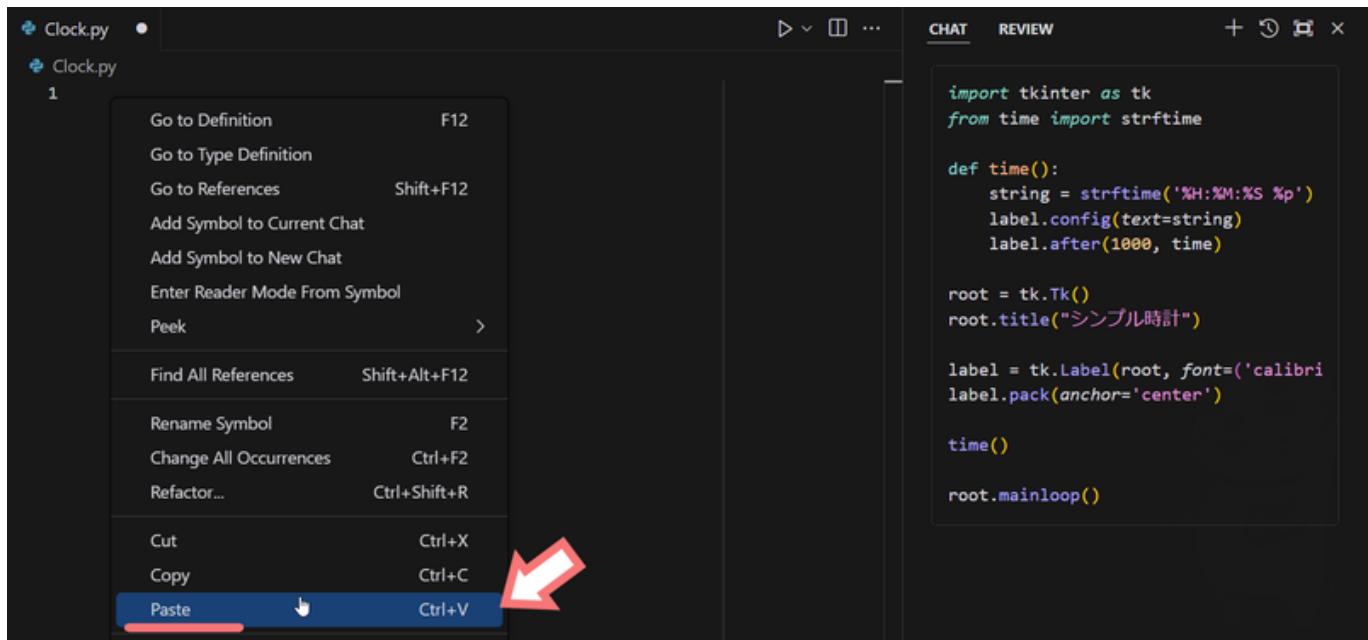
```
python
import tkinter as tk
from time import strftime

class ClockApp:
    def __init__(self, root):
        self.root = root
```

中央のプログラムを書き込むエリアに、プログラムを貼り付けます。

右クリック -> Paste してください。

二章 最初のチャレンジ！



```
import tkinter as tk
from time import strftime

def time():
    string = strftime('%H:%M:%S %p')
    label.config(text=string)
    label.after(1000, time)

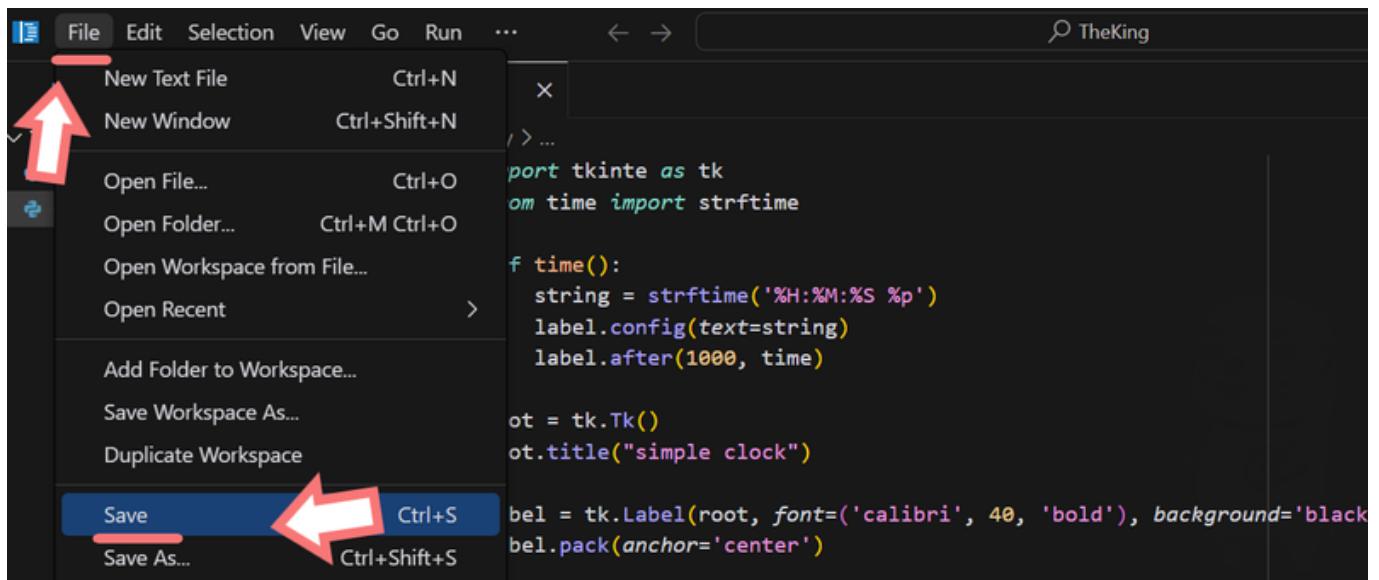
root = tk.Tk()
root.title("シンプル時計")

label = tk.Label(root, font=('calibri', 40, 'bold'), background='black')
label.pack(anchor='center')

time()

root.mainloop()
```

プログラムを貼り付けたら File -> Save しておきましょう



```
port tkinter as tk
om time import strftime

f time():
    string = strftime('%H:%M:%S %p')
    label.config(text=string)
    label.after(1000, time)

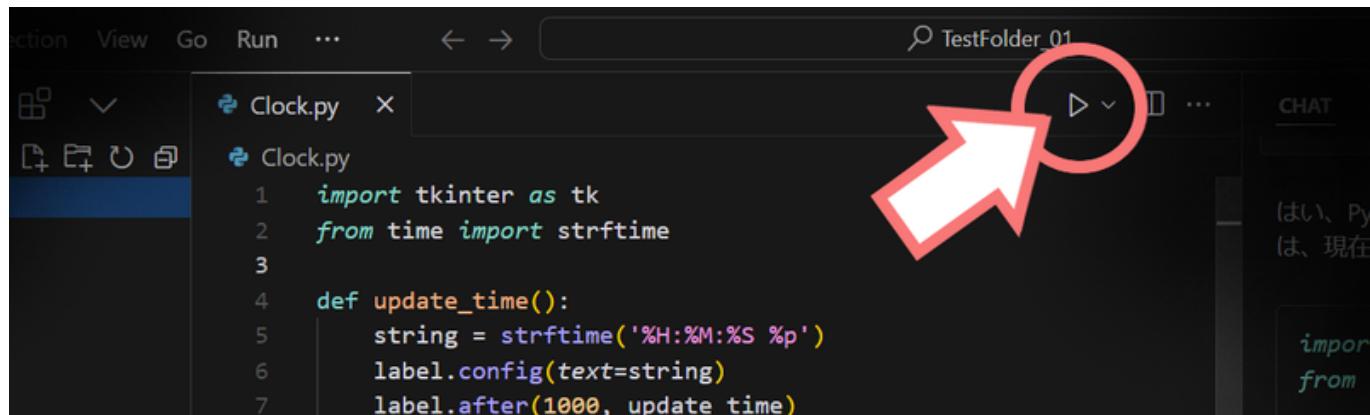
ot = tk.Tk()
ot.title("simple clock")

bel = tk.Label(root, font=('calibri', 40, 'bold'), background='black')
bel.pack(anchor='center')
```

♦ プログラムを実行する

ついに！実行の時です！

cursorの **実行ボタン▶** を押し、プログラムを動かしましょう！



```
action View Go Run ... ⏪ ⏩ TestFolder_01 CHAT  
Clock.py x  
Clock.py  
1 import tkinter as tk  
2 from time import strftime  
3  
4 def update_time():  
5     string = strftime('%H:%M:%S %p')  
6     label.config(text=string)  
7     label.after(1000, update_time)
```



実行ボタンをポチっと…



simple clock
19:01:18 PM



えっ・・・



わああああああああああ！！

できちゃった！！！！

すごーい！！

皆さんのところも時計のウィンドウが表示されましたか？

もしもうまくいかなかったら



うわーん！
なんかごちゃごちゃしてきて良く分からなくなつた！

♦ cursorを再起動してみよう

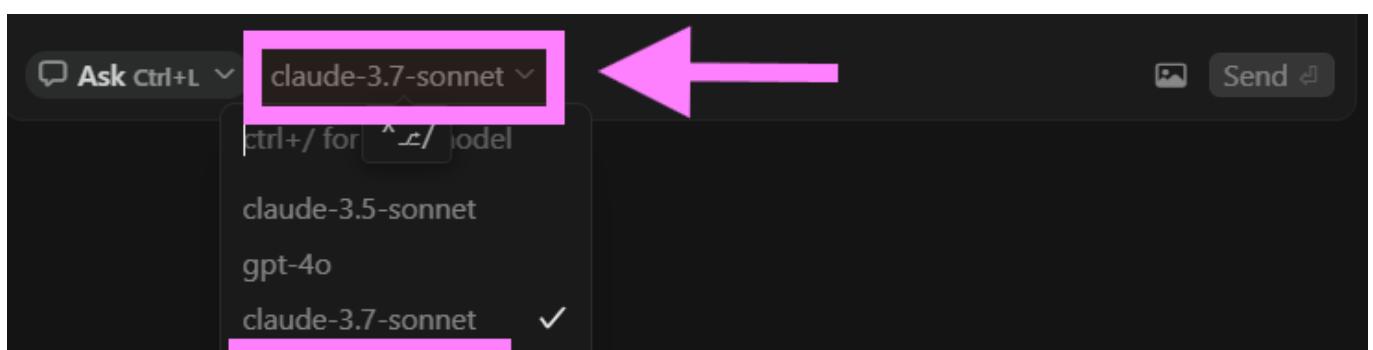
Cursorがうまく動かないときは、一度cursorを閉じて再起動してみましょう。これで多くの問題が解決することができます。

♦ Cursorに相談してみよう

Cursorはプログラムのことだけじゃなくて、いろんなことを教えてくれます。分からぬことがあったら、気軽に聞いてみましょう！きっと役に立つヒントをくれるはずです。

♦ モデルを変更してみよう

お返事がヘンテコで困ったら、チャット欄を選択して、モデル選択欄から 今度は、**claude-3.7-sonnet** を選んでみましょう。



プログラムをカスタマイズしてみよう

次に、この時計をカスタマイズしましょう！



色をオレンジにしたい
こだわりが出ちゃう

では、追加の修正依頼を送りましょう！**追加のお願いは、右下にチャット欄がありますよ！**

```

File Edit Selection View ... ← → ⌂ TheKing
THEKING Clock copy.py Clock.py < > CHAT REVIEW + ⌂ ×
import tkinter as tk
from time import strftime
# Ctrl+L to chat, Ctrl+K to generate
def time():
    string = strftime('%H:%M:%S %p')
    label.config(text=string)
    label.after(1000, time)

root = tk.Tk()
root.title("simple clock")

label = tk.Label(root, font=('calibri', 40),
label.pack(anchor='center')

time()

root.mainloop()

```

```

import tkinter as tk
from time import strftime

def time():
    string = strftime('%H:%M:%S %p')
    label.config(text=string)
    label.after(1000, time)

root = tk.Tk()
root.title("シンプル時計")

label = tk.Label(root, font=('calibri', 40),
label.pack(anchor='center')

time()

root.mainloop()

```

Ask followup (Ctrl+Shift+Y), 1 to select

gpt-4 chat ctrl+D codebase

♦ お願いの書き方

@ ♦Clock.py

色を変えてみて！オレンジ色がいいな！

v gpt-xx v

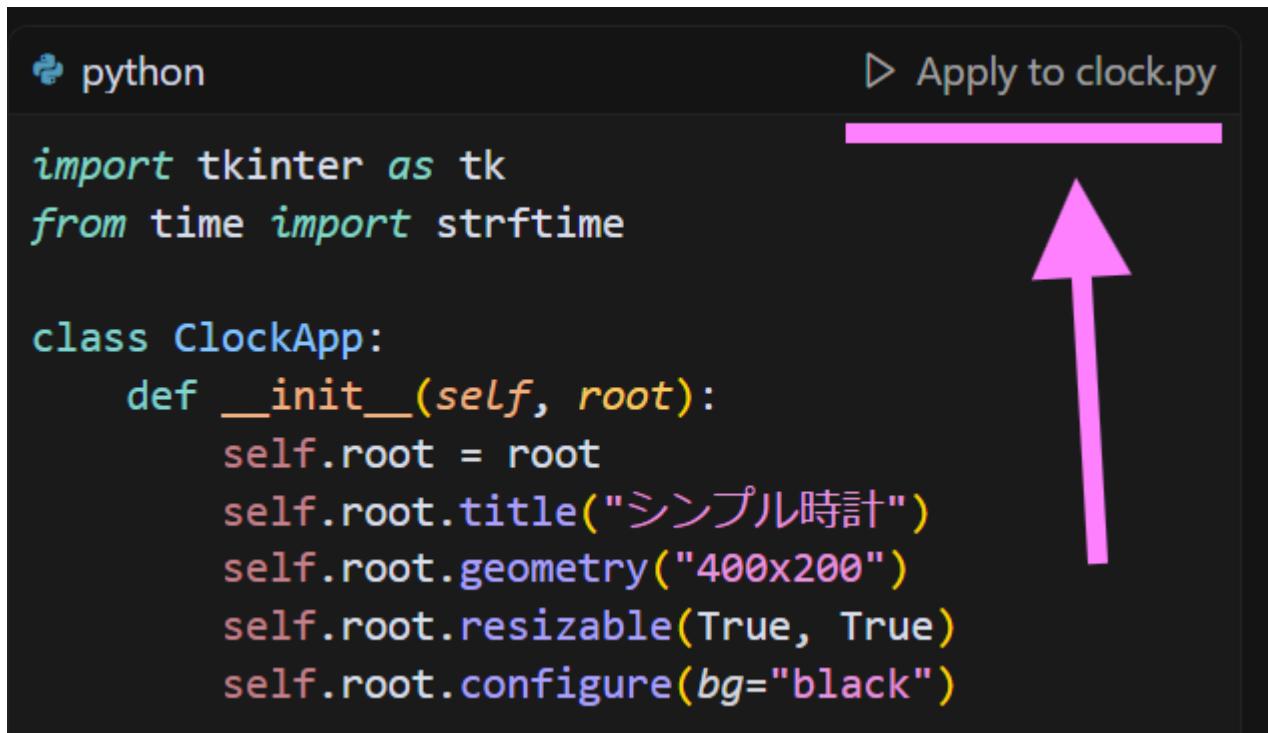
□ Send ↴

エンターを押してcursorにプログラムを書いてもらいましょう！

♦ 修正を自動適用する

cursorが修正案を書いてきたら、これをエディタに反映しましょう。全部手でコピペ…
をする必要はありません！

マウスカーソルを、お返事のプログラム部分に近づけていって、**Applyを押す**と、
cursorが自動的にコードを適用してくれます。



```
python > Apply to clock.py
import tkinter as tk
from time import strftime

class ClockApp:
    def __init__(self, root):
        self.root = root
        self.root.title("シンプル時計")
        self.root.geometry("400x200")
        self.root.resizable(True, True)
        self.root.configure(bg="black")
```

ボタンを押すとcursorがプログラムを更新しはじめます。

中央の画面が書き換わっていくのがみえるはずです。



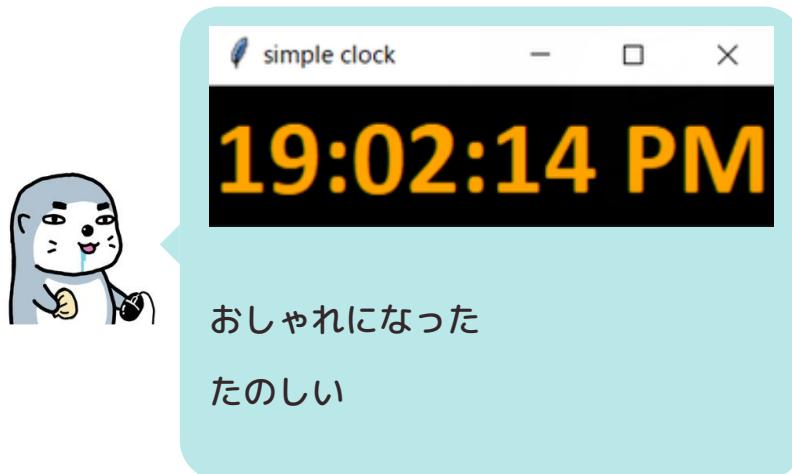
日付のラベルを作成
日付のラベルを作成 - オレンジ色に変更
`self.date_label = tk.Label(
 root,
 font=('Helvetica', 18),
 background='black',
 foreground='white'
 foreground='#FFA500' # 少し明るいオレンジ色
)`

Accept all

Accept

緑の部分はcursorの修正提案です。この状態は、まだ提案しているだけで、修正はされていませんから、あなたが **Acceptを押して確定** してください。Acceptを押すまでは、変更は反映されません！！

最後に**実行▶**して結果を確認します。

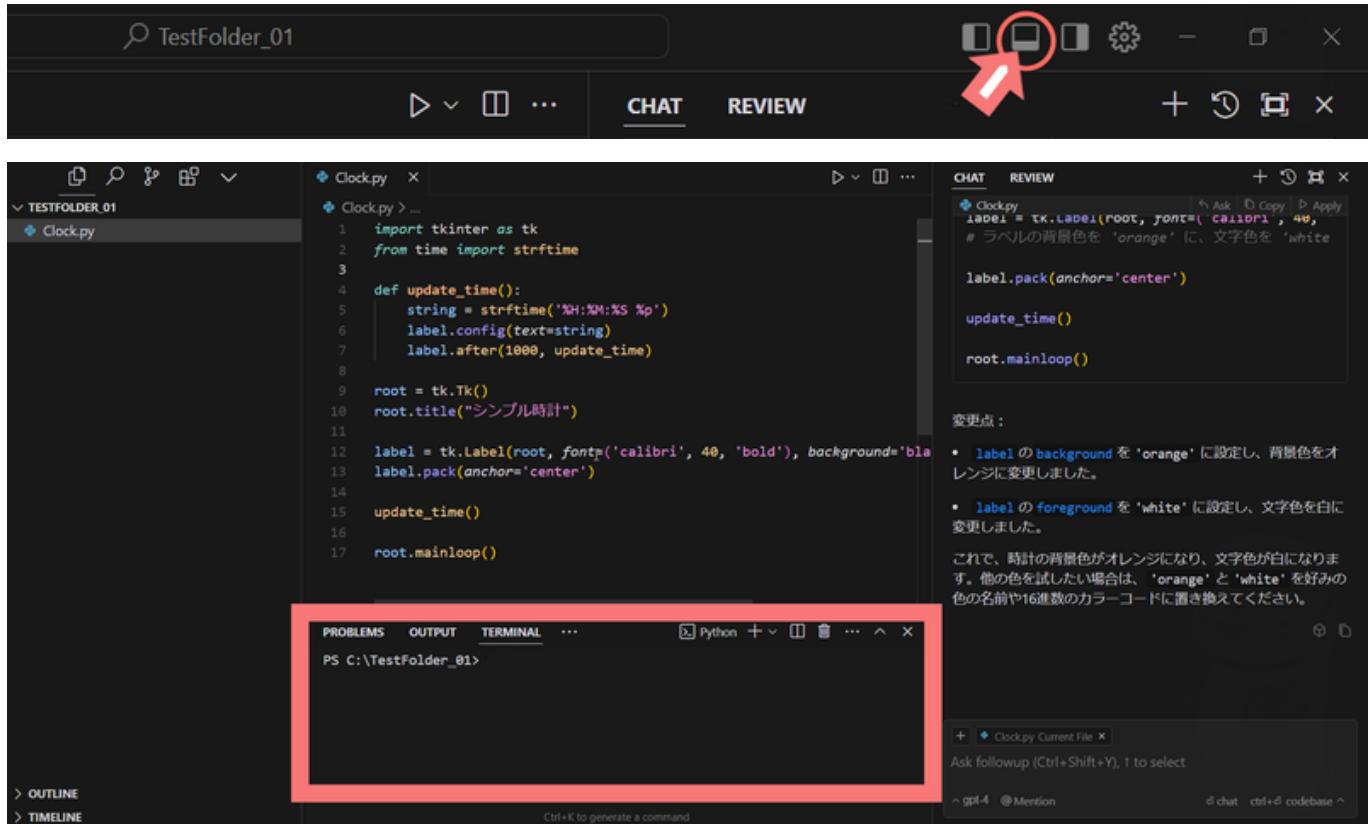


このように、自分のイメージ通りのツールになるまで、cursorとやりとりを繰り返していくことになります。

エラーに対応してみよう

プログラムが動かなかったらcursorと会話しながらエラーを直していきます。

① こちらのボタンを押しエラー表示画面(ターミナル)を出します。



② ここではためしに、"tkinter" と書いてあるところを "tkinte" に書き換えるなど、正しい部分をわざと間違えて書き直し実行してみましょう！

```
Clock.py  X
Clock.py > {} tk
1 import tkinte as tk
2 from time import strftime
3
```

実行して、時計が表示されなくなったら、エラーが発生しています！

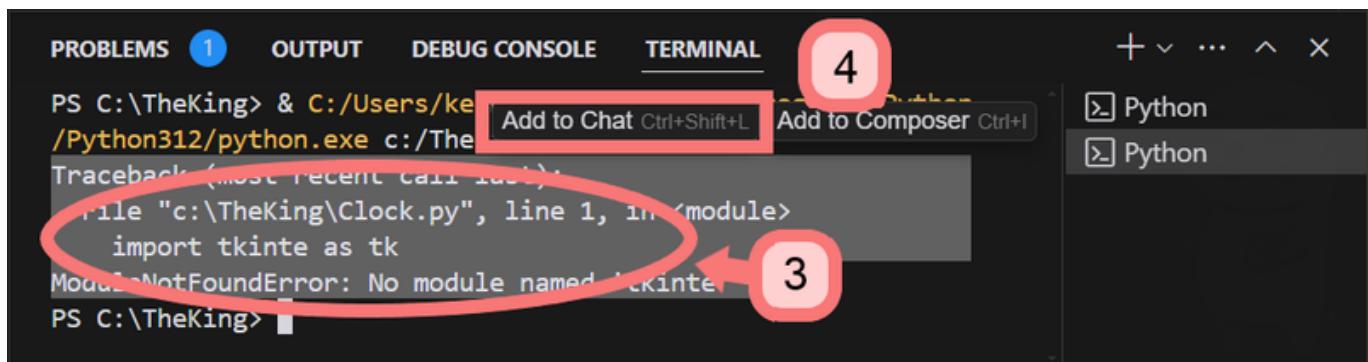
エラー表示画面(ターミナル)にはずらっと文字が出てきているはずです。

```
hon312/python.exe c:/TestFolder_01/Clock.py
Traceback (most recent call last):
  File "c:/TestFolder_01/Clock.py", line 1, in <module>
    import tkinte as tk
ModuleNotFoundError: No module named 'tkinte'
PS C:/TestFolder_01>
```

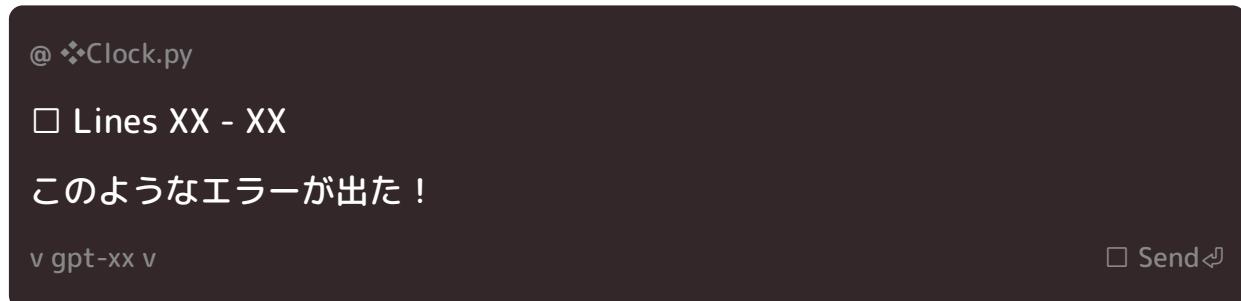
Ctrl+K to generate a command

③ 「Error」という文字が見えたたら、エラーの説明をドラッグして選択する（分からぬ場合は全て選択）

④ 「Add to Chat」を押して、選択した内容をチャット欄に送る



⑤ エラーが出たことを書き添えてエンターを押し、修正を依頼しましょう！



修正案をcursorが送ってくるので、Applyを使ってエディタに適用してください。

二章 最初のチャレンジ！



あれ？ぼくのところは、"✓Continue"って出てるけど…？

Applyを今開いているファイルに対して行っていいの？と確認されている状態です
ので、✓Continueを押して続行してください



continueするぜ！

その後、Acceptをして、縁の提案が消えたら、実行▶してください！
エラーは消えていますか？



ふたたび時計があらわれた！安心！

もしこここまでで躓いてしまったら、cursorに質問しながら、手順を繰り返してみてください。

ツールをデスクトップから起動しよう



つくった時計をデスクトップから起動したい！

いいですね！cursorに頼んでいきましょう！

♦ お願いの書き方

@ ♦Clock.py

ぼくの時計、Clock.pyをデスクトップから起動できるようにしたい！どうすればいいの？初心者向けに短く教えて！

Ask claude-3.7-sonnet v

Send ↗

♦ cursorのお返事

一番簡単な方法は、Pythonスクリプト (.py) をバッチファイル (.bat) から呼び出すことです。以下の手順で実現できます：

1. デスクトップに新しいテキストファイルを作成し、名前を「Clock.bat」に変更します。
2. このバッチファイルを編集し、以下の内容を記述します：

```
@echo off
python "C:\path\to\your\Clock.py"
pause
```

3. C:\path\to\your\ の部分を、実際の clock.py ファイルがある場所のパスに置き換えてください。

これで、デスクトップの clock.bat をダブルクリックするだけで、Pythonスクリプトが実行されます。



返事の内容が全然違う…

cursorはお返事の内容が毎回違います。

ここに書いてあるのはあくまで例であって、あなたの画面に出てくる返事は違うのです。
ここではいったん、「batを作る方法」をやってみましょう。



フォルダは"C:\path\to\your\Clock.py"じゃないと思うけど…
ファイルがどこにあるのかわからない…

ファイルの場所を調べたい場合は

左画面のファイル名の上で、右クリック -> Copy Path

または 右クリック -> Reveal in Explorer



ようし！かきかえて。。。えいっ！



ダブルクリックしたら起動した！

これだけは覚えておこう！

困ったら思い出して！

♦ コピー・貼り付けのショートカット

「Ctrl」+「C」でコピー

「Ctrl」+「V」で貼り付け

右クリックしても何も出ないときはこれを使おう

それくらい知ってるし



♦ 改行の方法

・チャット欄で改行がしたい場合は、「Enter」ではなく

「Shift」+「Enter」

えっ…



三章 べんりなツールを作ろう

大量の作業を自動化して楽をしよう

完成するもの

- ・ファイルリスト作成ツール
- ・エクセルのリストをチェックするツール

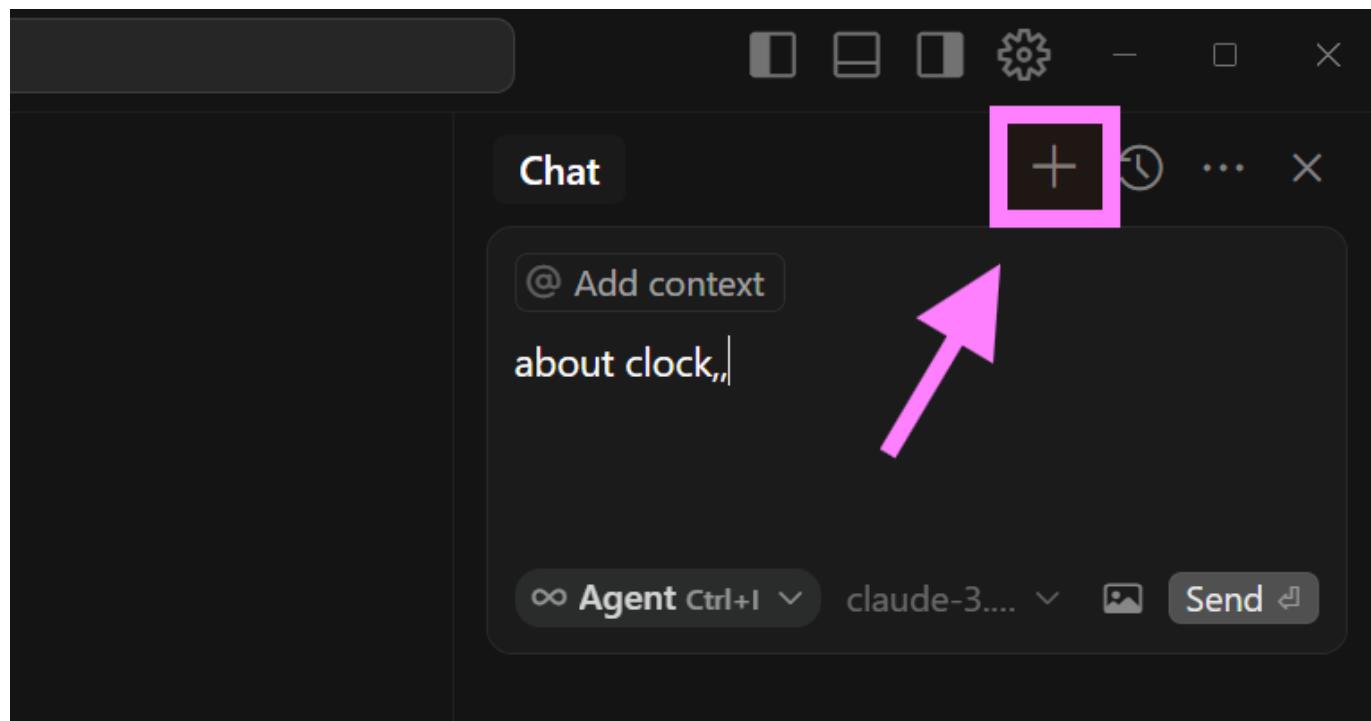
どんな犠牲を払ってでも楽をしたい



New Chat から始めよう

新しい依頼を送る前に、**チャット欄をリセット** しましょう。

下記の+ボタンを押してみてください。



チャット欄がリセットされたでしょうか。

cursorとのチャットを長くやり取りし続けると、冒頭に入力した情報をずっと踏まえて回答しようとして、cursorが混乱してしまうので、**こまめに押す** ようにしましょう。

ファイルリストツールを作ろう



上司がファイルリスト作れって…
ファイル1000個あるのに…

それではcursorに相談してみましょう！

フォーマットが決まってる場合は、サンプルをつけてあげるといいですよ！

♦ お願いの書き方

@ Add context

フォルダの中のファイルリストを作るツールをつくって！ フォルダの名前と、ファイルの名前と、ファイルサイズが書いてあるリストです！ こんな感じで！

#例:

```
C:\Osama\Images\  
--- Image001.png(100KB)  
--- Image002.png(12KB)
```

Ask claude-3.7-sonnet v

Send ↴

♦ cursorのお返事

フォルダ内のファイルリストを作成するPythonスクリプトを作成しましょう。このスクリプトは、指定されたフォルダ内のファイルとそのサイズを表示します。以下のコードを使用できます：

```

import os

def get_file_size(file_path):
    size_bytes = os.path.getsize(file_path)
    if size_bytes < 1024:
        return f"{size_bytes}B"
    elif size_bytes < 1024 * 1024:
        return f"{size_bytes // 1024}KB"
    else:
        return f"{size_bytes // (1024 * 1024)}MB"

def list_files(folder_path):
    print(f"{folder_path}")
    for item in os.listdir(folder_path):
        item_path = os.path.join(folder_path, item)
        if os.path.isfile(item_path):
            size = get_file_size(item_path)
            print(f"--- {item}({size})")

# 使用例
folder_path = r"C:\Osama\Images" # ここにリストを作成したいフォルダのパスを入力してください
list_files(folder_path)

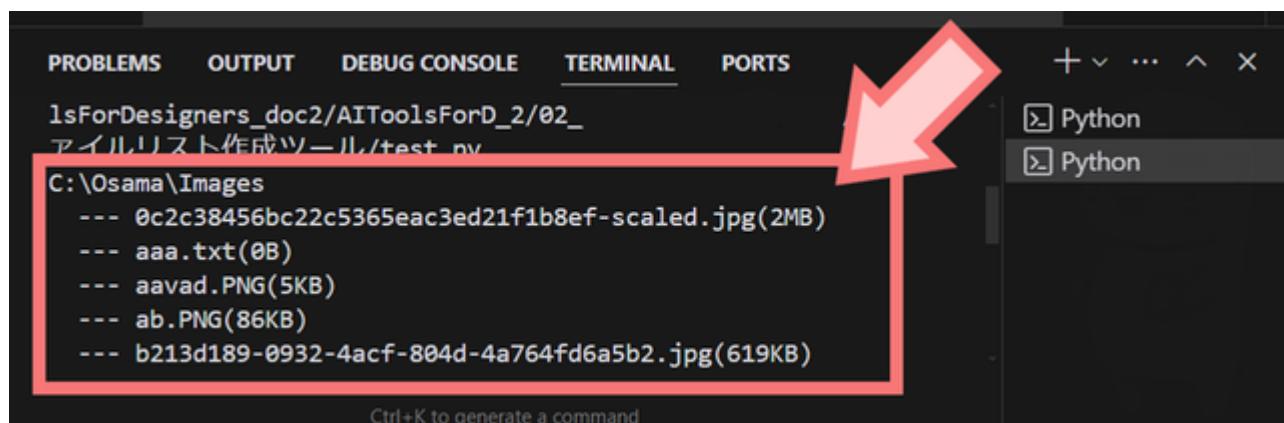
```

♦ 実行してみる

早速結果を見ましょう！

list_test.pyを作って、プログラムを貼り付けたら、**実行ボタン▶** を押してください。

そして、下のターミナル画面を見てみましょう！



```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS + ... ^ X
lsForDesigners_doc2/AIToolsForD_2/02_
アイルリスト作成ツール/test.py
C:\Osama\Images
--- 0c2c38456bc22c5365eac3ed21f1b8ef-scaled.jpg(2MB)
--- aaa.txt(0B)
--- aavad.PNG(5KB)
--- ab.PNG(86KB)
--- b213d189-0932-4acf-804d-4a764fd6a5b2.jpg(619KB)

```

The terminal window shows the execution of the `list_files` function, listing files in the `C:\Osama\Images` directory with their sizes. A red arrow points from the terminal output area to the right sidebar, which displays available Python environments.



テキストファイルに保存してほしいなあ

♦ 様々なカスタマイズ

では、直してもらいましょう。頼んでは返事をApplyしていってください。

@ ♦list_test.py

結果はテキストファイルに出力してよね！

Ask claude-3.7-sonnet v

Send ↗

@ ♦list_test.py

ウィンドウUIもつくって！

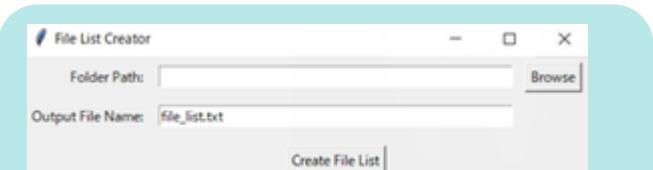
Ask claude-3.7-sonnet v

Send ↗

♦ 完了



ファイル1000個が一瞬でおわった…！



見た目をオシャレにしよう



UIもっとかわいくしたいな～

では、かわいくなるようにお願いを送りましょう！

ここでは **PySide6**というライブラリを使います。



えっ…PySide6ってなに？

いったん気にせずに下記のように送ってみてください！

♦ お願いの書き方

@ ❁list_test.py

このスクリプトをPySide6でオシャレなゆめかわ色にして

👉 Ask claude-3.7-sonnet v

□ Send ↴

◆ cursorのお返事

はい、PySide6を使用して、おしゃれなゆめかわ色のGUIに変更しましょう。

以下に、変更後のコードを示します。

```
import os
from PySide6.QtWidgets import QApplication, QWidget, QVBoxLayout,
QHBoxLayout, QLabel, QLineEdit, QPushButton, QFileDialog,
 QMessageBox
from PySide6.QtGui import QColor, QPalette
from PySide6.QtCore import Qt
.
.
.
省略
```



はいコピペして実行！
あれっ？ エラーだよ？

```
ModuleNotFoundError: No module named 'PySide6'
```



何をいってるんだろう？
cursorに聞いてみよう

♦ お願いの書き方

@ Add context

ModuleNotFoundError: No module named 'PySide6'

って言われたけどどうしたらいいの？

初心者だからわかりやすく短く教えてね

v gpt-xx v

submit ↴ ctrl + ↴ codebase

♦ cursorのお返事

このエラーは、PySide6がインストールされていないことを示しています。解決するには：

1. コマンドプロンプトを開きます。
2. 次のコマンドを実行します：

```
pip install PySide6
```

これでPySide6がインストールされ、エラーが解消されるはずです。



pipってなに？

pipは、Pythonに新しい機能を追加するためのツールです。

普段は使えない特別な機能や拡張機能（ライブラリ）を簡単にインストールして、プログラムに取り入れることができます。

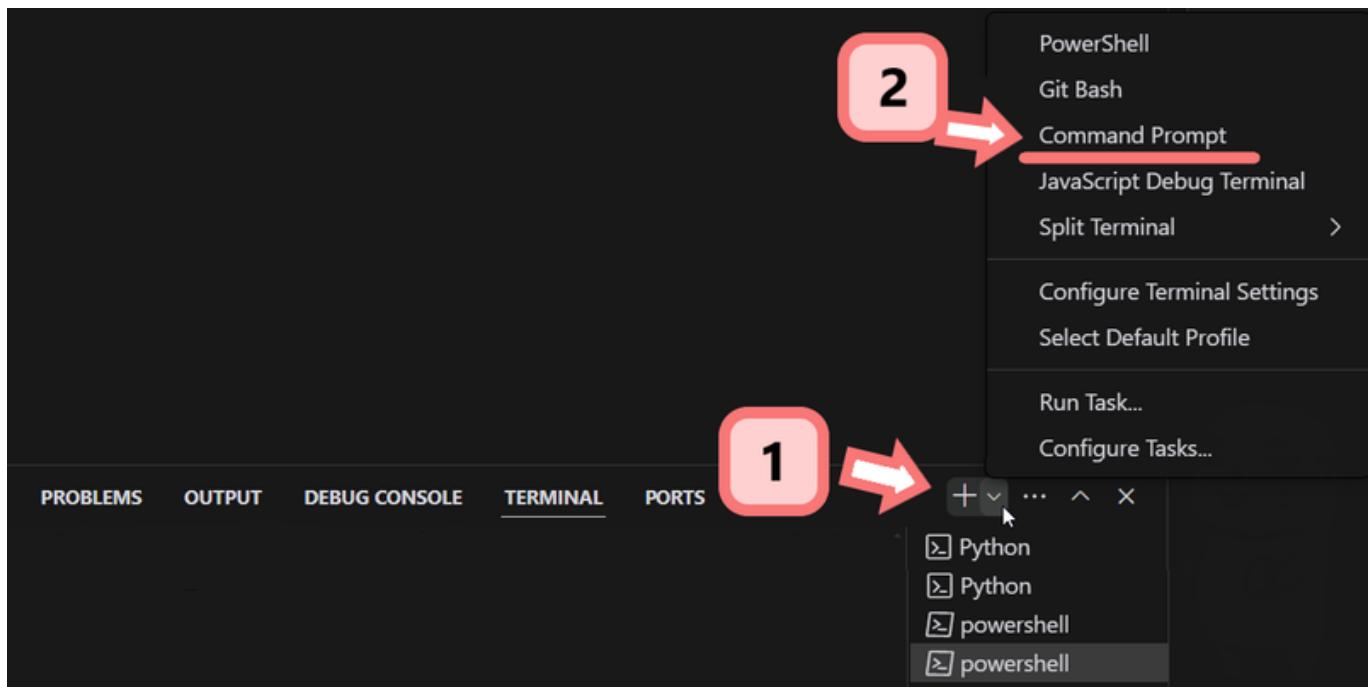
かわいいUIにしたい場合は入れないとダメです！



めんどくさいけど
かわいくしたいからしょうがないね

◆ ターミナルに入力をする

ターミナルの画面でCommand Promptを開きます。



そうしたらターミナル内にコマンドを入力してインストールをしますよ！

```
pip install PySide6
```

これを **ctrl + C** でコピーし、ターミナルで **ctrl + V** で貼り付けてください！

The screenshot shows a Windows terminal window with the following text:

```
Microsoft Windows [Version 10.0.26100.3194]
(c) Microsoft Corporation. All rights reserved

C:\test\test>pip install PySide6
```

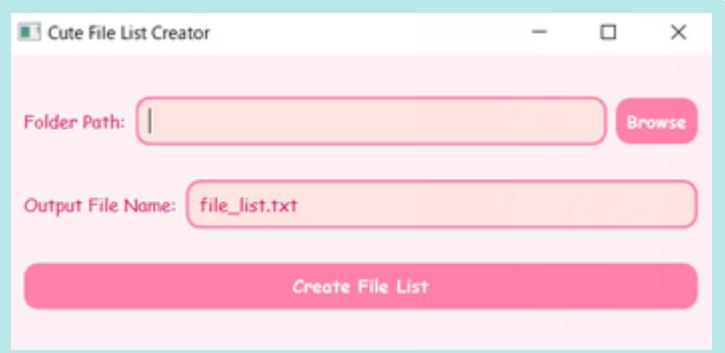
エンター を押して実行したらインストールが始まります。



何かがインストールされて行く…

インストールが終わったら、もう一度さっさく作ったPythonファイルを **実行▶** してみましょう。

♦ 完了



めちゃかわいいじゃん

PySide6やるじゃん

エクセルチェックツールを作ろう



上司がまた無茶を言ってきた！
エクセルのリストとファイル名を見比べてチェックしろだって！
ファイル1000個あるのに！

では、エクセルからデータを出力してもらいましょう！

♦ pipをする

今回は先にpipをして機能を追加します！

```
pip install pandas openpyxl
```



ターミナルにぶちこむ！

♦ お願いの書き方

@ Add context

エクセル(C:\TheFile.xlsx)のリスト(列の名前はFileName)と、
C:\Osama\フォルダの中のファイル名を見比べて、
フォルダにエクセルのファイルがちゃんとあるのかチェックしたい！
何かいい方法はないですか？

Ask claude-3.7-sonnet v

Send ↗

♦ cursorのお返事

はい、Pythonを使ってこのタスクを実行する良い方法があります。以下のスクリプトは、Excelファイルからファイル名のリストを読み取り、指定されたフォルダ内のファイルと比較します。

```
import os
import pandas as pd

excel_path = r'C:\TheFile.xlsx'
df = pd.read_excel(excel_path)
excel_files = set(df['FILE_Name'].tolist())
folder_path = r'C:\Osama'
folder_files = set(os.listdir(folder_path))
missing_files = excel_files - folder_files
extra_files = folder_files - excel_files
print("Excelリストにあるが、フォルダにないファイル:")
for file in missing_files:
    print(f"- {file}")
print("\nフォルダにあるが、Excelリストにないファイル:")
for file in extra_files:
    print(f"- {file}")
print(f"\n合計: {len(missing_files)} 個のファイルが見つかりません。")
print(f"合計: {len(extra_files)} 個の追加ファイルがフォルダに存在します。")
```

このスクリプトを実行する前に、以下のライブラリをインストールする必要があります： pip install pandas openpyxl



はいはい

"exceltest.py"にコピペして実行！



ターミナルに答えが出てる！助かる！

♦ さらにカスタマイズ

@ ♦ExcelCheck.py

エクセルファイルは指定したいな！

Ask claude-3.7-sonnet v

Send ↗

@ ♦ExcelCheck.py

UIつくって！PySide6を使ってゆめかわいい感じで！

Ask claude-3.7-sonnet v

Send ↗



PySide6をおぼえたぞ

♦ 結果



できちゃった



こんなツールも作れます

他にもこんなツールが簡単に作れます。

- エクセルファイルをXX形式に変換するツールを作って！
- カラーピッカーをつくって！
- ToDoのツールをつくって！
- ファイルのバックアップツールを作って！
- (作ったツールを)exeにして！

試しにいろいろ頼んでみてくださいね。

ゆめ いっぱい！



四章 知識を吸収させてみよう

知識を吸収させてから回答してもらおう

完成するもの

- ・プレゼン資料

ツールじゃないの？



最初に

♦ ソフトウェアごとの独自言語

それぞれのソフトウェアには独自の「言語」があります。例えば：

- ・ウェブブラウザにはhtml
- ・ExcelにはVBA
- ・MayaにはMEL

これは、地域によって言葉が違うのと同じようなものです。

♦ 苦手な言語への対応

cursorには得意な言語と苦手な言語があります。苦手な言語を書かせる場合は、**事前に知識を吸収して答える機能**を活用して対応します。この章ではその方法を説明します。

どのようなソフトウェアでも、基本的に対応方法は同じなので、ここでは基礎知識として手順をなぞってもらえればと思います。

Marpを使ってプレゼン資料を作ろう

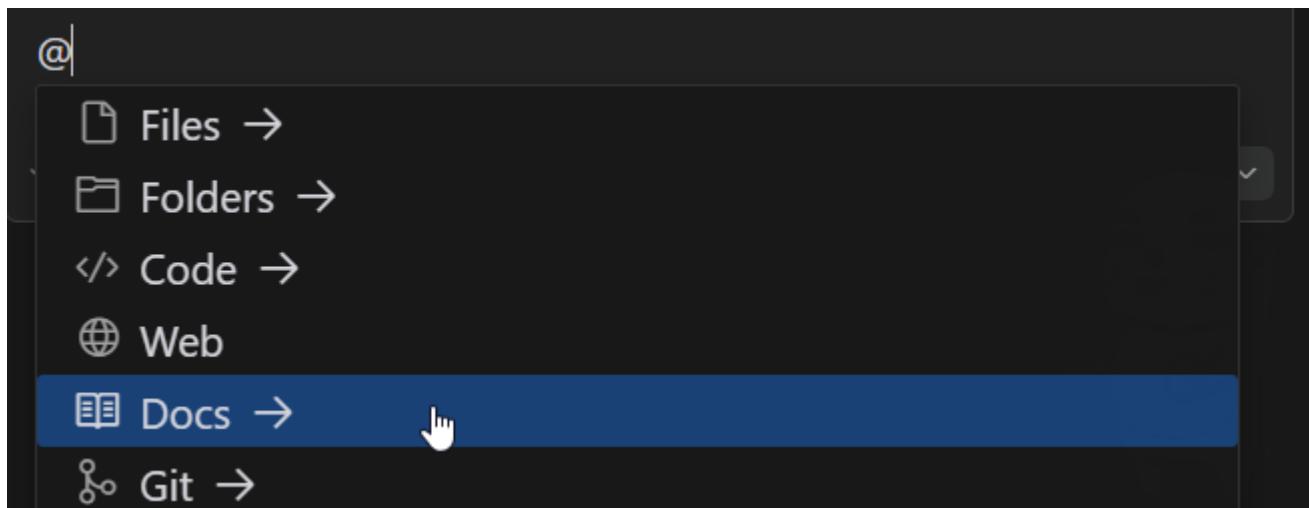


おやつをしまう冷蔵庫を買ってもらいたいなあ

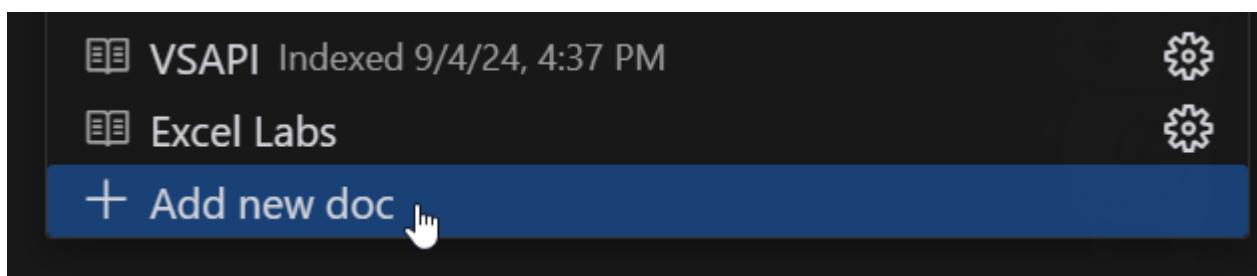
プレゼン用フォーマットの専門知識として、ここでは「Marp」の知識を吸収してもらいます。

◆ cursorに新しい知識を吸収してもらう

- ① チャット欄に「@」と入力するか、チャット内の@AddContextを押して一覧を表示。「Docs」を選択します。



- ② 「+ Add New Doc」を選択します。

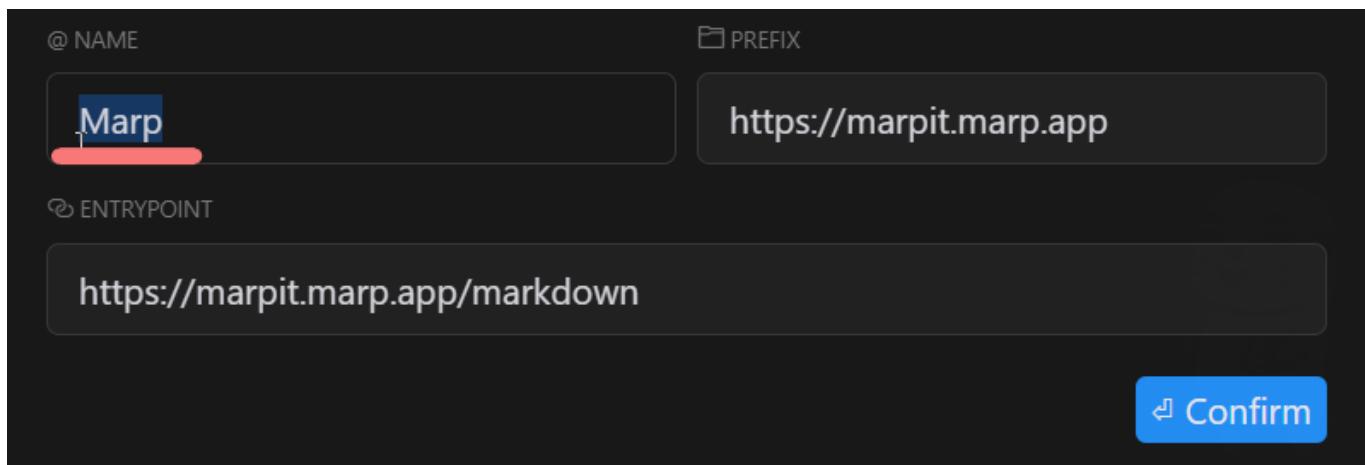


③ Marpの資料が掲載されたホームページのアドレスを入力します。

入力するアドレス：<https://marpit.marp.app/markdown>



④ 好きな名前をつけて **confirm** します。ここでは **Marp** と名付けましょう。



お疲れさまです。cursorが情報を吸収しました。



めんどいなあ…毎回やるとか言わないよね？

毎回は必要ありません。

一回情報を吸収したら、その後は 「@Marp」 と入力することでいつでも呼び出して使えます。

♦ 吸収した知識を元にしてお願ひをする

- 先ほどつけた名前 **@Marp** を先頭に付けてお願ひを送信しましょう。
- (内容は自分らしく書き換えてくださいね)

@ Marp

会社でプレゼンしたいから、いい感じにまとめて Marp書式にして！

#プレゼンしたい内容

おやつはおいしいという話！なんかね おやつっておいしいでしょ！いざってときのためにちょっといいおやつを 冷蔵庫の中に入れておくと、嫌なことがあっても俺にはおやつがある！というつよいきもちをもっていられるから いいよということがいいたい！だから結論としてオフィスにいい感じの冷蔵庫をおいてほしいんですよ

v gpt-xx v

□ Send ↗

これでcursorは知識を吸収した状態でお返事を書いてくれます！

marp: true

theme: default

おやつは…(省略)

お返事の内容を **MarpTest_01.md** という名前で保存します。

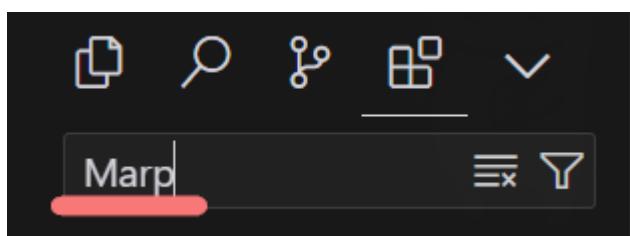
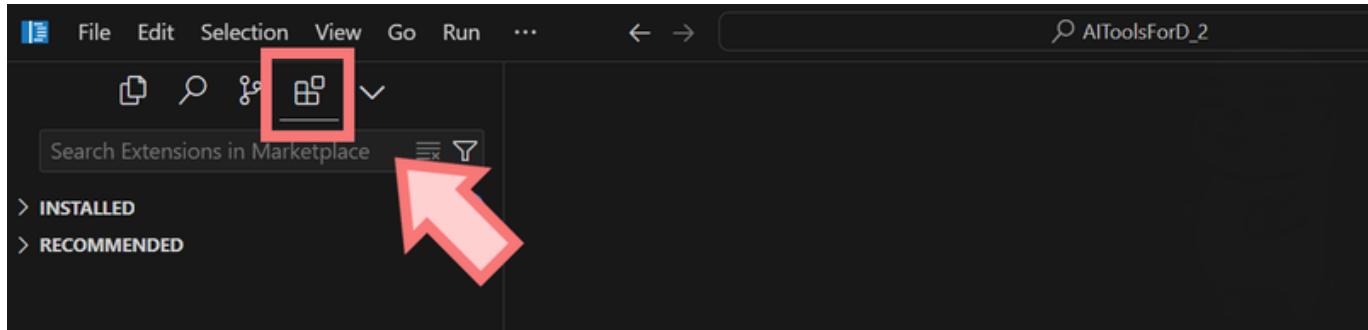


こんどは".py"じゃないんだね

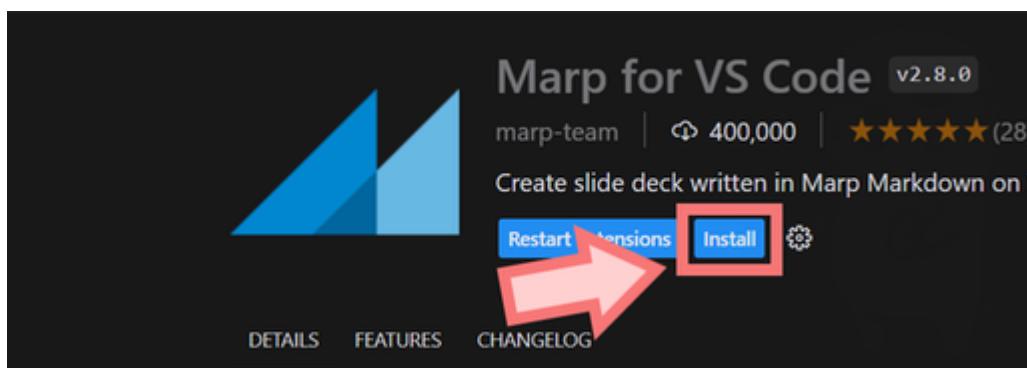
♦ 実際のフォーマットで表示してみる

cursorにMarpをインストールします。

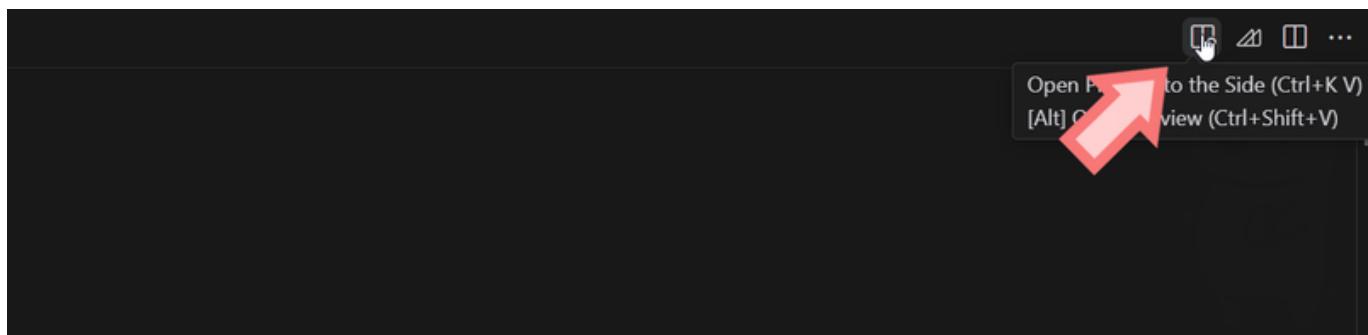
① 拡張機能のアイコンを押して、「Marp」で検索します。



② 見つかったらインストールして下さい。



③ インストール後、Marpで表示するボタンを押します。



右側にプレゼン用のフォーマットが表示されました！



A screenshot of a code editor window titled "test3.md". The left pane shows the Markdown file content:

```
03_実践テスト > Marp > test3.md > Preview test3.md X ...  
1  ---  
2  marp: true  
3  theme: default  
4  class:  
5    - lead  
6    - invert  
7  ---  
8  
9  # おやつはおいしいという話  
10  
11 ---  
12  
13  ## おやつの魅力  
14  
15  おやつっておいしいでしょ！いざというときのために、ちょっといい  
  おやつを冷蔵庫に入れておくと、嫌なことがあっても「俺にはおやつ  
  がある！」という強い気持ちを持っていられます。  
16
```

The right pane is titled "Preview test3.md" and displays the rendered presentation slide. A large red arrow points from the text "右側にプレゼン用のフォーマットが表示されました！" to the preview pane. The preview shows the text "# おやつはおいしいという話".



あらやだ！

あっという間にプレゼン資料ができちゃった！

◆ 内容の加工

- ・ 見た目をカスタマイズしましょう

@ ▼MarpTest_01.md 📄 Marp

ゆめかわいくして！

v gpt-xx v

□ Send ↴

◆ 結果

🌈 The Charm of Snacks 🌈

Snacks are delicious, aren't they? If you keep some nice snacks in the fridge for when you need them, even when bad things happen, you can maintain a strong mindset like "I have snacks!"



すごいかわいい

♦ プログラム以外のお願いもできます



やったー！がっつり頼んじゃお！！！

@ ▼MarpTest_01.md 📄 Marp

もっと上司が冷蔵庫を買いたくなるようにしてください！！！

v gpt-xx v

□ Send ↴

♦ 結果

💼 Business Benefits of an Office Refrigerator

- 🔍 Increased employee satisfaction and retention
- 🔍 Improved workplace nutrition and health
- 🔍 Enhanced team bonding and social interactions
- 🔍 Reduced time spent on external food runs

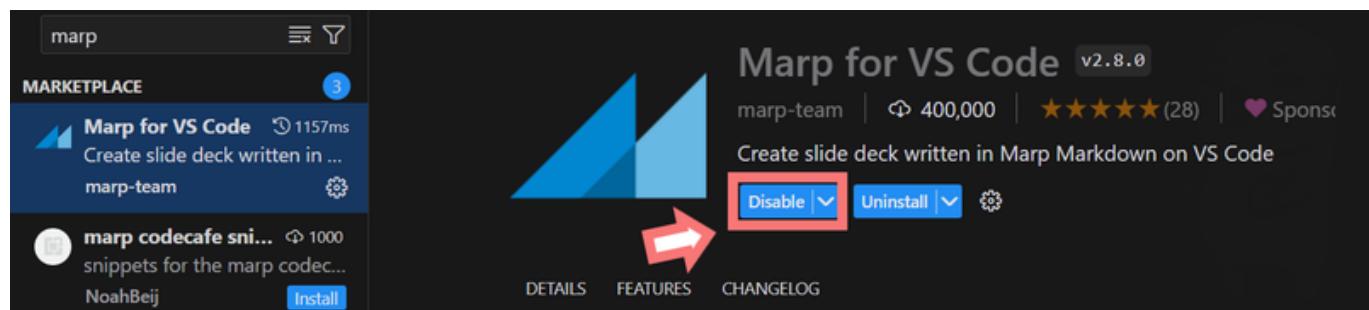


そうそう…そういう事を言いたかったんだよ

@を使って知識を吸収してもらったことで、このような色々な仕様に対応できるようになります！

♦ 使い終わったらMarpを無効化する

使い終わったら、拡張機能画面にいって、Marpを選択し、「Disable」を押して無効化しましょう。



知識の吸収方法いろいろ

♦ 個人的ランキング

第一位 : Docs

ご存じDocs機能です。URLを入れると階層下にあるものを一式取得してくれます。

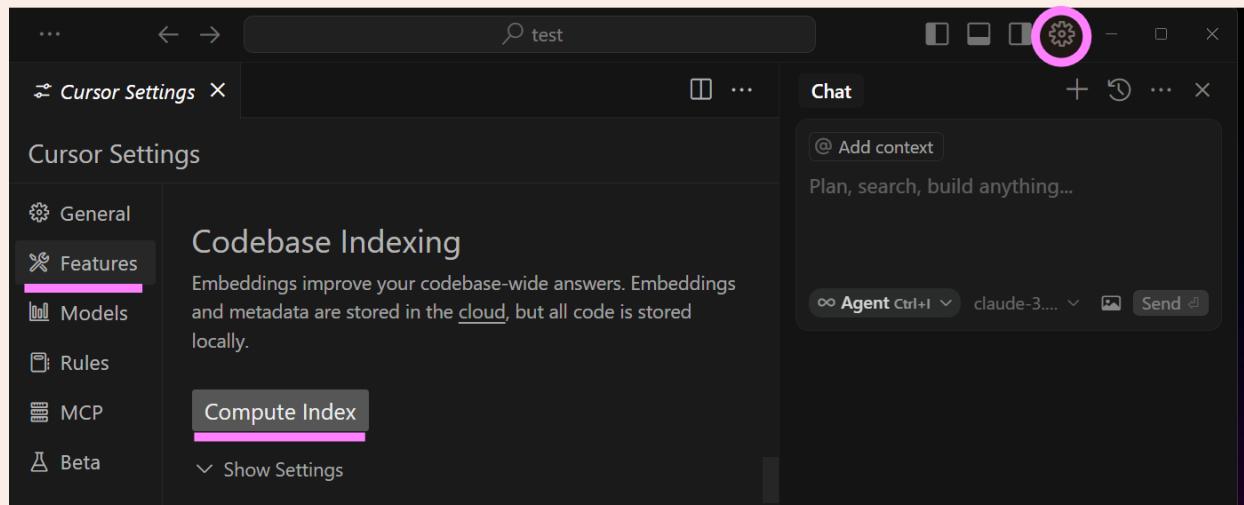
第二位 : Codebase:

開いているフォルダ以下の階層全体を参照して回答します。

Docsの上限ページ数が1000ページなので、それ以上のページ数を参照させたい場合は、ローカルにファイルを保存してこれを使います。また、Docsがうまく読み取れない場合、ウェブページやPDFをテキスト形式で保存して、Codebaseを使います。

利用には予めCodeBaseIndexingの作業が必要です。

設定ボタン⚙ -> Features -> CodebaseIndexing



これを押すことで、階層下の中身をCodebaseという知識袋みたいなものに格納しておくことができます。以後は、@codebaseすることで、知識袋から知識を引き出せるようになります。フォルダの中身が変わったら更新しなおしてください。

第三位：<https://~>

URLをはりつけると、そのウェブページを参照して回答します。1ページだけ見てほしいときはこれが便利です。

♦ たまに使う機能

Terminals: ターミナルを読んでくれる機能

Git: Gitを利用している人向けの機能。@commitを選んで、コミットメッセージ書いてと頼むとコミットメッセージを生成してくれます。

Web: ウェブ全体を検索し、cursorが自動的に情報を収集します。収集したデータをもとに回答します。当たりはずれが大きすぎてあまりうまく使っていません。

過度の期待は禁物



ようし！大量の情報を吸収させちゃおっと！

ちょっと待ってください！

LLMは多量の文字を一度に処理できないため、関連がありそうな部分を探して回答しています。決して全文を理解してくれるわけではないのです！

また、LLMは情報が多いと思考力が落ちるので、大量の知識を吸収させて解答させると良い結果になるというわけではなく、できれば少ない知識を与えた方が精度は上がります。



めんどくさい…悲しい…