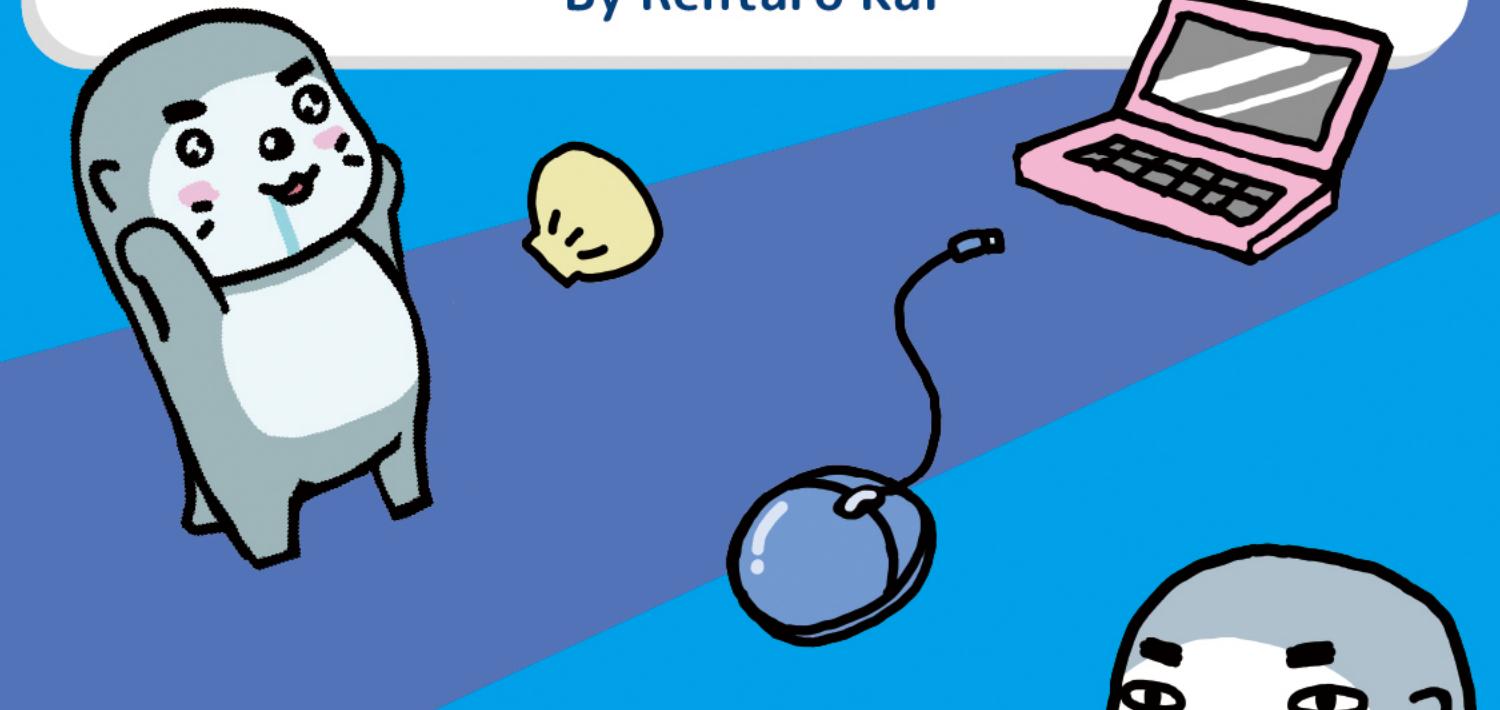


Hotate Press

# Beginner's CURSOR

By Rentaro Kai



Now! For all!  
Let's strive together!

Automate & boost efficiency!

Build your tools!

Photoshop, Maya, Blender...etc



backimg



Manman the Sea Otter Ventures onto Land



I'm Otteroo!



Just being cute? Nah, I've got ambition!



Let's study together!!

# Introduction

---

## ♦ About the Author

Rentaro Kai

Usually spends his time making various 3D stuff with Maya.

Just when he was drowning in endless digital chores, he discovered cursor, and thanks to its handy automation tools, life became dramatically easier!

He hopes this book serves as your first step into cursor and helps anyone who struggles as he once did.

Here's what some of Rentaro's friends had to say after he introduced cursor to them:

- Ever since cursor arrived, I've barely written any code myself.
- I even let cursor handle simple print statements.
- Honestly, cursor is better at naming files and variables than I am.
- Not paying for cursor genuinely disrupts my workflow.
- cursor keeps suggesting features I didn't even know existed.
- Even languages I don't know feel doable as long as cursor's around.

Pretty glowing reviews!



## ◆ Who This Book is For

Based on the above, this book is intended for people like:

- Designers who want to write scripts
- Anyone interested in automating routine tasks
- Those who've given up on programming books before
- People who already script a bit but want an easier way

Through learning cursor, this book aims to provide beginners with an enjoyable introduction to programming.

If you want to dive deeper into programming, we recommend exploring specialized books covering the fundamentals separately.

This book was written based on a Windows environment, but most operations should work similarly on Mac as well.



# Table of Contents

---

Introduction .....	5
--------------------	---

About the Author .....	5
------------------------	---

Who This Book is For .....	6
----------------------------	---

Table of Contents .....	8
-------------------------	---

Chapter 1: Install Cursor! .....	10
----------------------------------	----

What is cursor? .....	11
-----------------------	----

Let's Install .....	12
---------------------	----

Let's Launch Cursor .....	16
---------------------------	----

Chapter 2: The First Challenge! .....	20
---------------------------------------	----

First, let's open a folder .....	21
----------------------------------	----

Let's Make a Clock App .....	22
------------------------------	----

Save and Run the Program .....	24
--------------------------------	----

If Things Don't Go Well .....	28
-------------------------------	----

Let's Customize the Program .....	29
-----------------------------------	----

Let's Handle Errors .....	32
---------------------------	----

Launch the tool from the desktop .....	35
--	----

Remember This! .....	37
----------------------	----

<b>Chapter 3: Let's Create Handy Tools .....</b>	<b>38</b>
Let's Start from New Chat .....	39
Let's Make a File List Tool .....	40
Let's Make It Stylish .....	43
Let's Make an Excel Check Tool .....	48
You can make tools like these .....	51
<b>Chapter 4: Let's Absorb Knowledge .....</b>	<b>52</b>
First .....	53
Let's Create Presentation Materials with Marp .....	54
Various Ways to Absorb Knowledge .....	62
Don't Expect Too Much .....	64

# Chapter 1: Install Cursor!

---

Let's install Cursor and get everything ready!

What a pain! Might cry!



## What is cursor?

cursor is: A next-gen editor software that programs for you by chatting with you.

Main features:

- Learns the script you want it to write
- Automatically fixes errors
- Provides suggestions



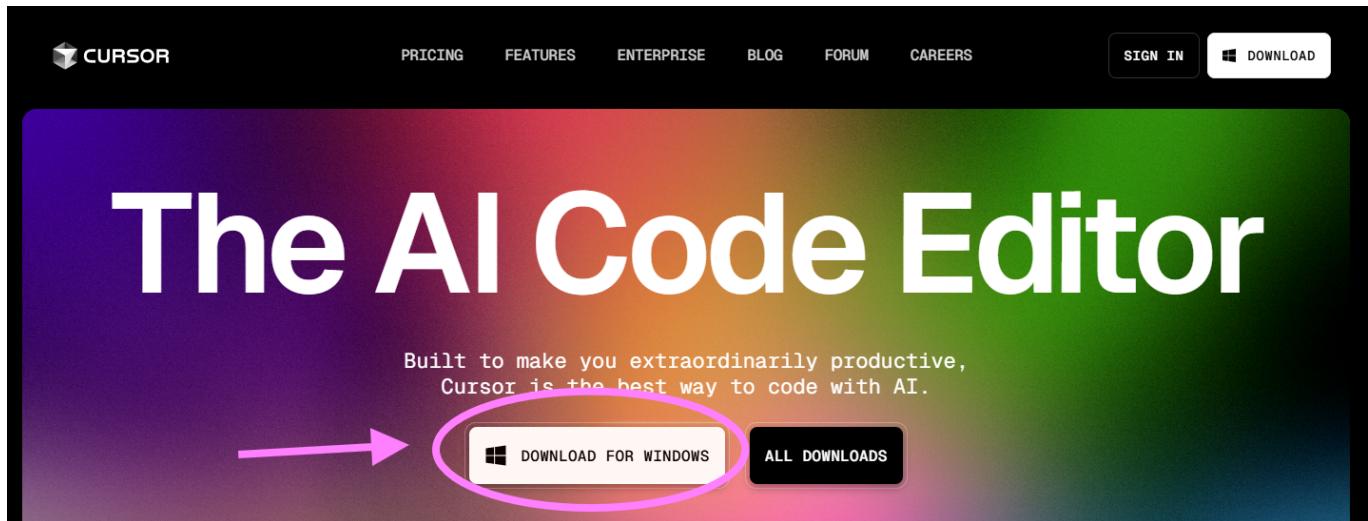
You can download it for free from the internet, and for the first two weeks, you can use the paid features as a trial (as of 2024). Let's get started right away!

## Let's Install

### ◆ Installing cursor

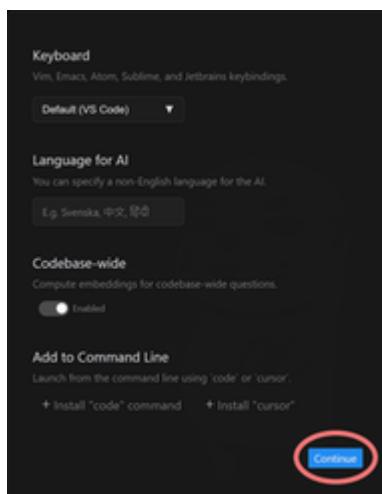
First, go to the official cursor website.

cursor Official Website · <https://www.cursor.com/>

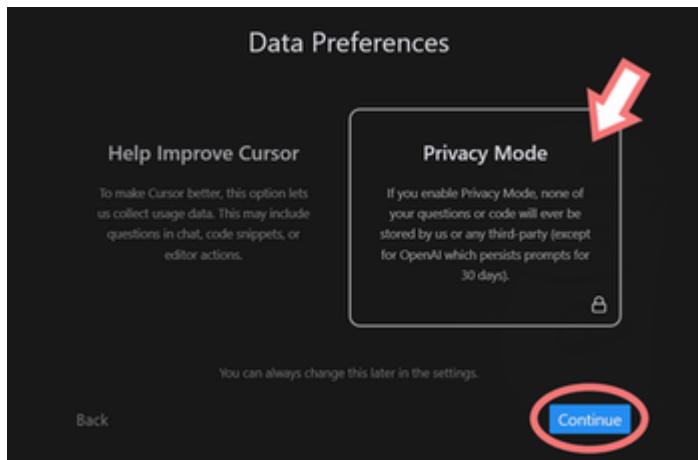


Click the Download icon to download the .exe file and run it.

On the first screen, just press continue with the default settings.



On the Data Preference settings screen, select Privacy Mode.



And that's it for installing cursor.

Next, let's install Python.

## ◆ Installing Python



What's Python!?

Isn't just cursor enough?

Python is absolutely, absolutely necessary! So hang in there and install it!



Ah, that Python. Yeah!

(I'll just pretend I get it for now)

# Chapter 1: Install Cursor!

Python Official Website · <https://www.python.org/>

The screenshot shows the Python official website at <https://www.python.org/>. 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 in a terminal window:

```
# Python 3: Fibonacci series up to n
>>> def fib(n):
    a, b = 0, 1
    while a < n:
        print(a, end=' ')
        a, b = b, a+b
    print()
>>> fib(1000)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987
```

To the right of the code, there's a section titled "Functions Defined" with a brief description of Python functions. Below the code, 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](#)".

Under the "Downloads" section, there are three main categories: "Get Started", "Download", and "Jobs". The "Download" section has 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 below.

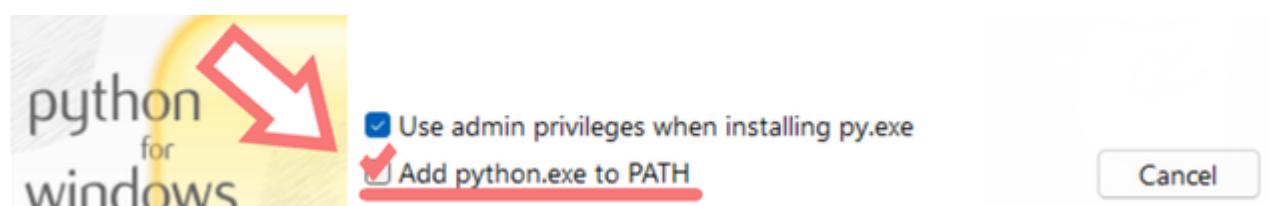
Version	Operating System	Description	MD5 Sum	File Size
Gzipped source tarball	Source release		d23d56b51d36a9d51b2b13d30c849d00	25.7 MB
XZ compressed source tarball	Source release		02c7d269e077f4034963bba6befdc715	19.5 MB
macOS 64-bit universal2 installer	macOS	for macOS 10.9 and later	78bd8d0795062b1df63e2b8d8386a5fa	43.5 MB
Windows installer (64-bit)	Windows	Recommended	bcbc2fcf9d739f776fb6414afc12c80d	25.3 MB
Windows installer (32-bit)	Windows		d151f5f116e11c4d40021527f51ddf67	24.0 MB

- ① Click the "Download" link for the "Latest" version
- ② Scroll down to the "Files" section and click the link labeled "Recommended" to download and run it.

A screenshot of the "Files" section of the Python download page. It shows a table with five rows. The fifth row, which corresponds to the "Windows installer (64-bit)" link, is highlighted with a red arrow pointing to it. The table columns are: Version, Operating System, Description, MD5 Sum, and File Size.

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
Windows installer (64-bit)	Windows	Recommended	bcbc2fcf9d739f776fb6414afc12c80d	25.3 MB
Windows installer (32-bit)	Windows		d151f5f116e11c4d40021527f51ddf67	24.0 MB

For the installation, most settings can be left as is, but make sure to check the box for **Add python.exe to PATH**.



Great job!

Now the installation is complete.

In the next section, we'll go over the basic usage of cursor.



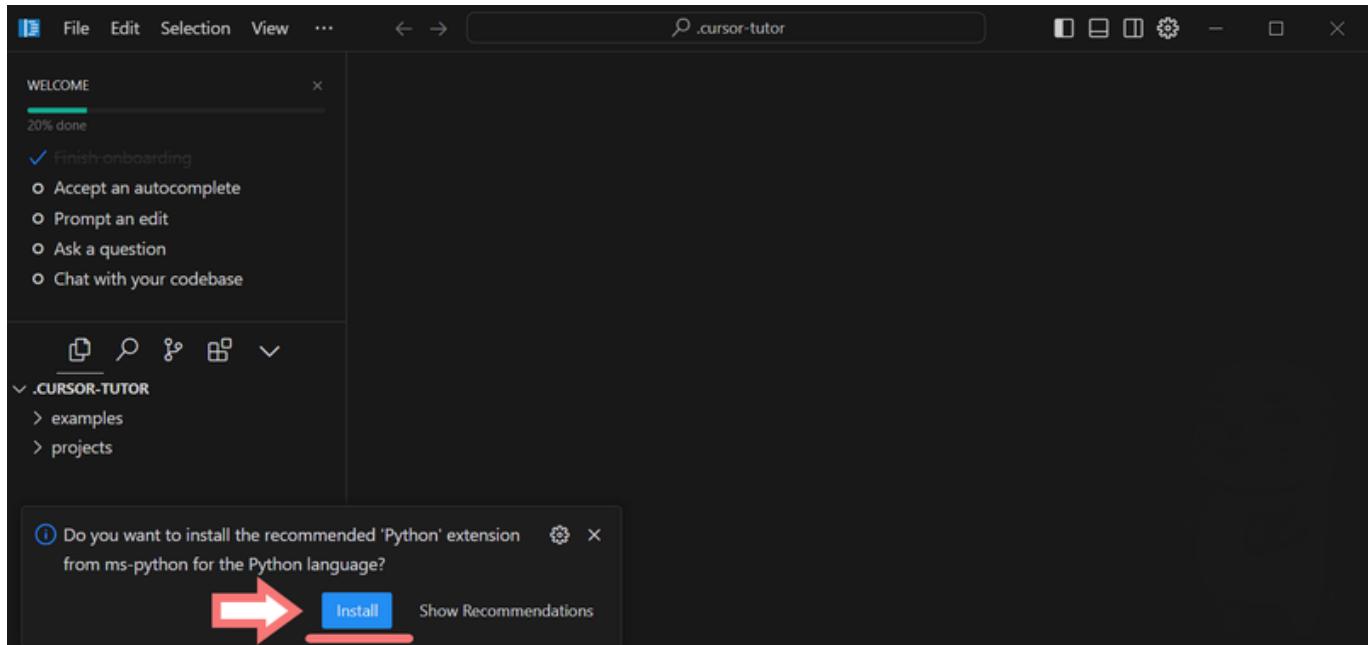
I tried super hard...!

Praise me lots!

## Let's Launch Cursor

### ♦ Display the Screen

Start up Cursor. You should see a screen like this:

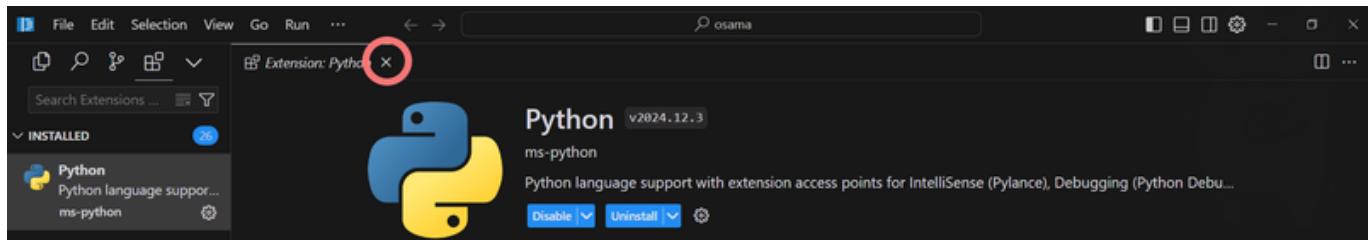


It's dark... scary...  
It's asking me to install something mysterious...!

"Python" is needed here too, so go ahead and click "Install"!



Whoa! Another dark screen... scary!



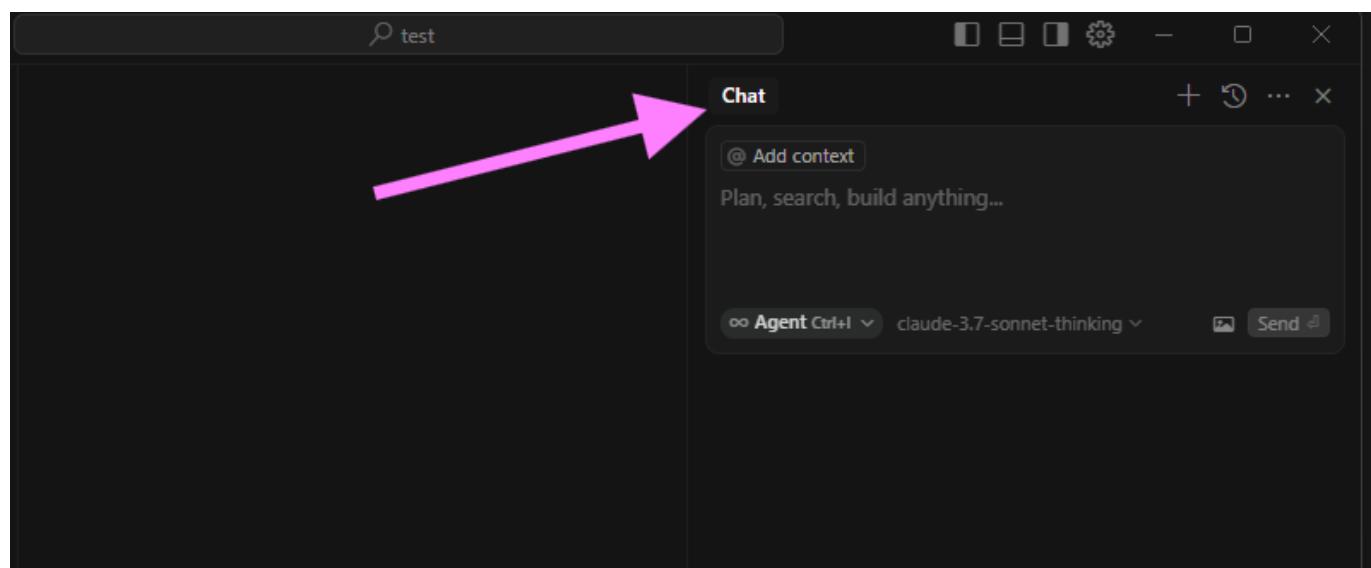
After installation, you'll see a screen like this. For now, just click the **x** to close it.

## ◆ Open the Chat Window

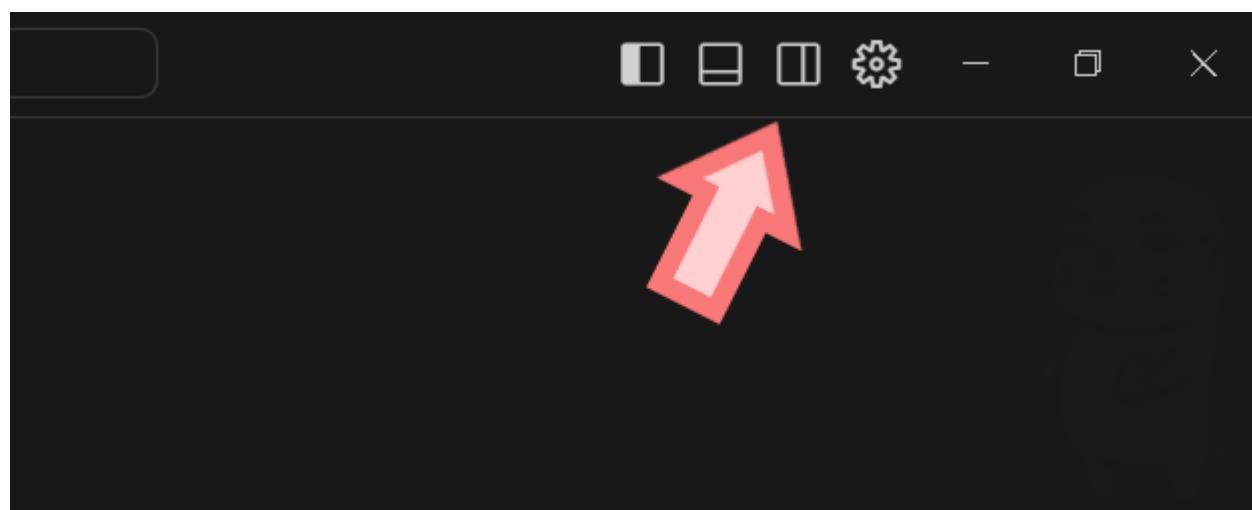
Cursor writes programs by chatting.

Let's get ready to chat.

If you see a Chat window on the right side of the screen, you're all set!



If the window isn't there, click the **□ icon at the top right** to display Cursor's chat window.

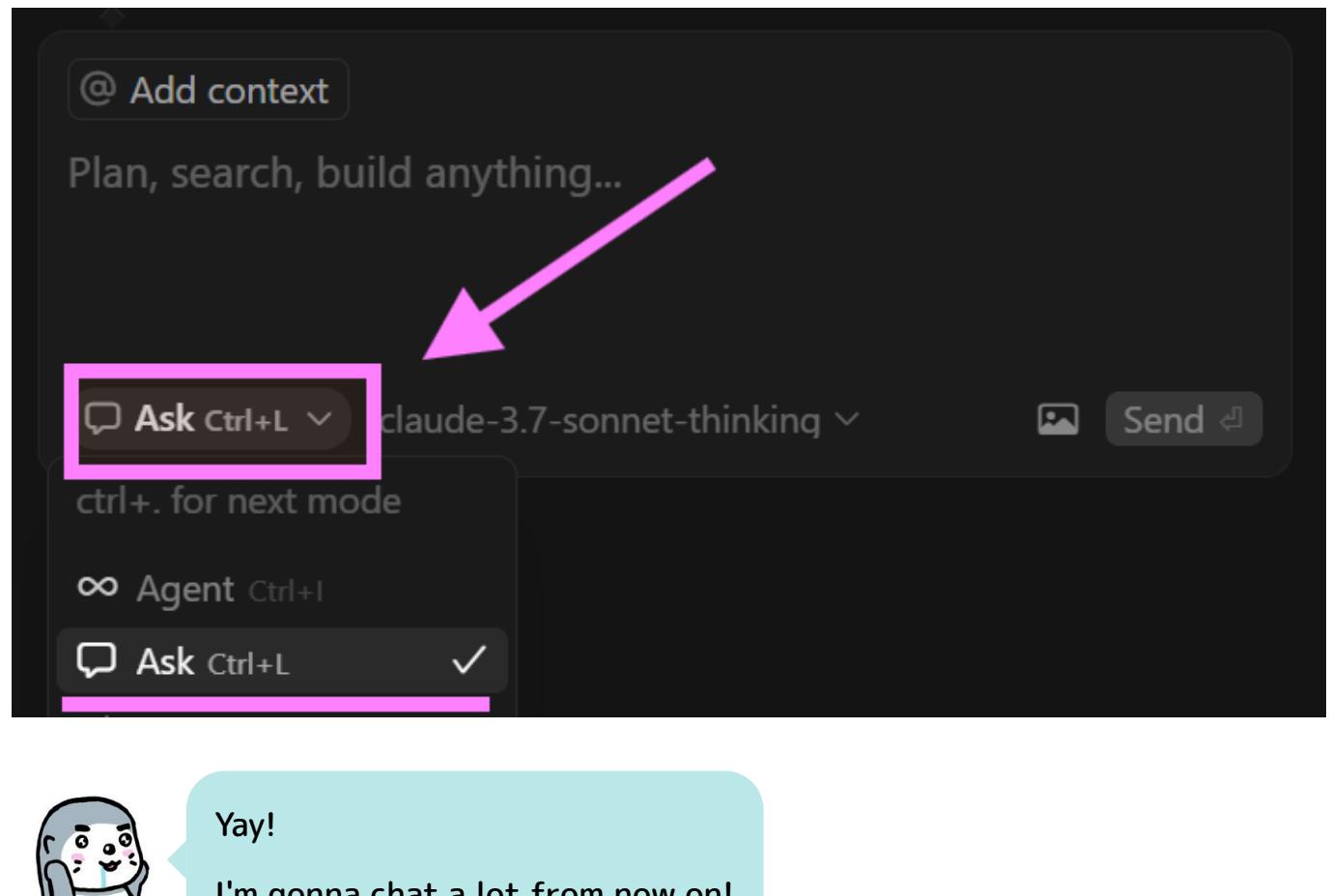


## Chapter 1: Install Cursor!

Next, the bottom left of the chat window should initially say "Agent."

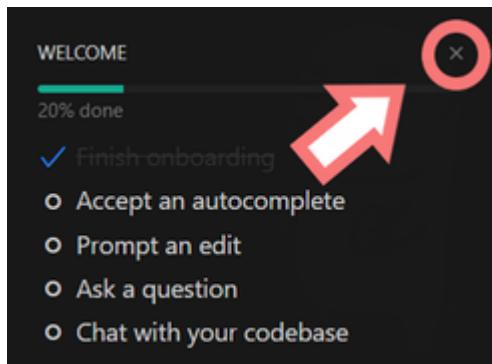
Change it to Ask (formerly CHAT).

We'll be using Ask for chatting for a while.



## ◆ You Can Close the Welcome Screen

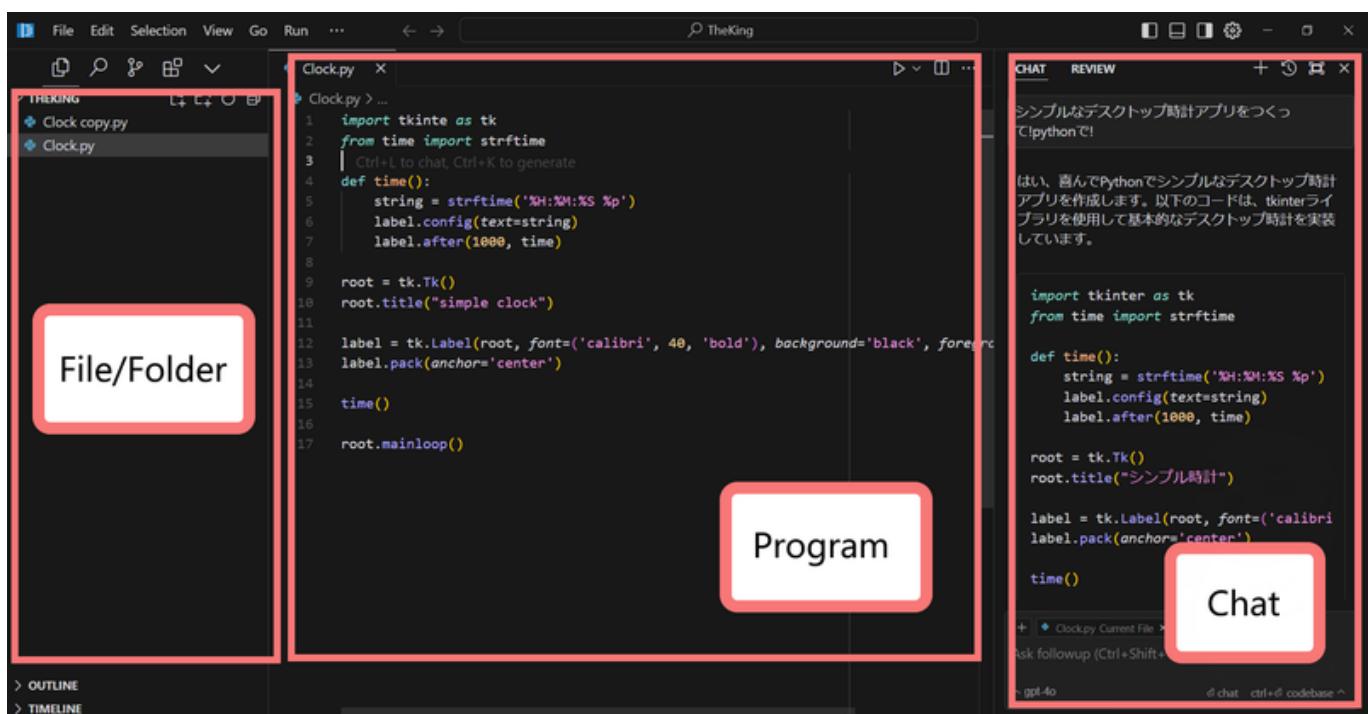
If the welcome screen pops up, feel free to close it.



## ◆ Screen Explanation

The screen is divided into three main areas:

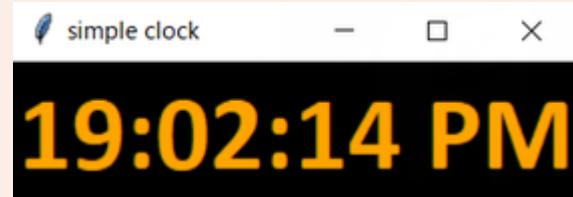
1. Left Panel: Displays a list of files and folders.
2. Central Editor: Create and edit programs here.
3. Right Chat Window: This is Cursor's chat interface.



# Chapter 2: The First Challenge!

Let's have cursor write a program!

What you'll create



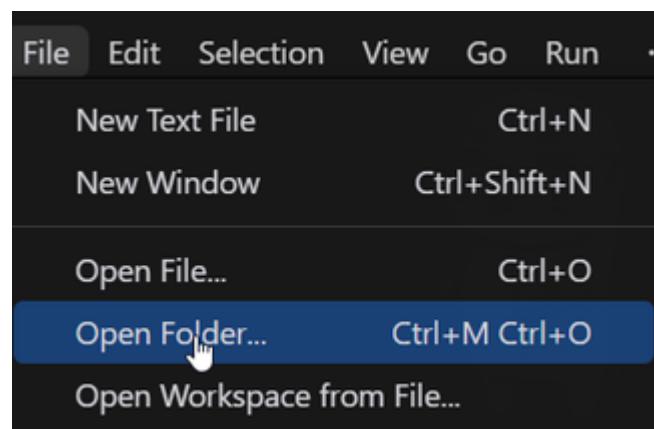
I don't really need a clock though...?



## First, let's open a folder

**cursor works on a folder basis.**

Select File -> Open Folder to open a folder.



Before you start working, create an empty folder with a name like "ClockFolder" in a convenient location, and select that.

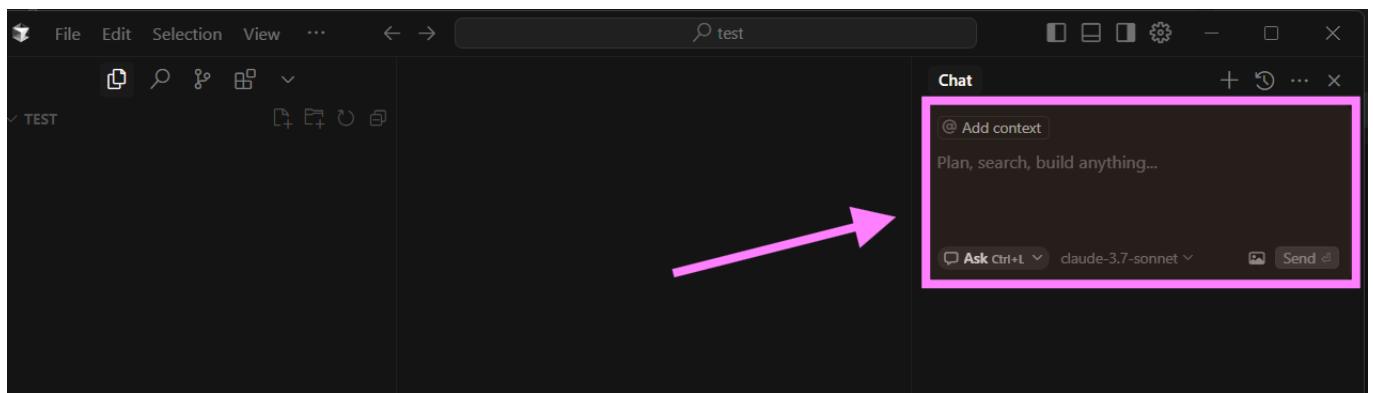
We're going to make an awesome clock in this folder!

### Let's Make a Clock App

#### ♦ Writing the Request

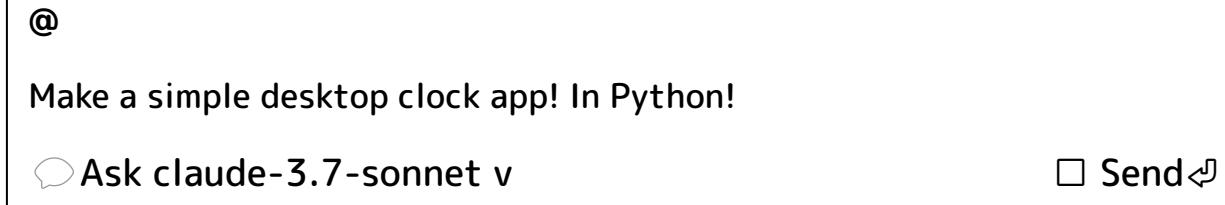
Let's write a request asking to create a clock.

First, select the chat box.



Then, write the request like the example below.

You don't have to ask in exactly the same way, as long as the meaning is clear.



#### ♦ cursor Writes the Code

After writing the request, press **Enter**, and cursor will start writing the code!

Sure! I'll happily create a simple desktop clock app using Python. Here's the code that implements a basic desktop clock using the tkinter library.

```
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("Simple Clock")

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

time()

root.mainloop()
```



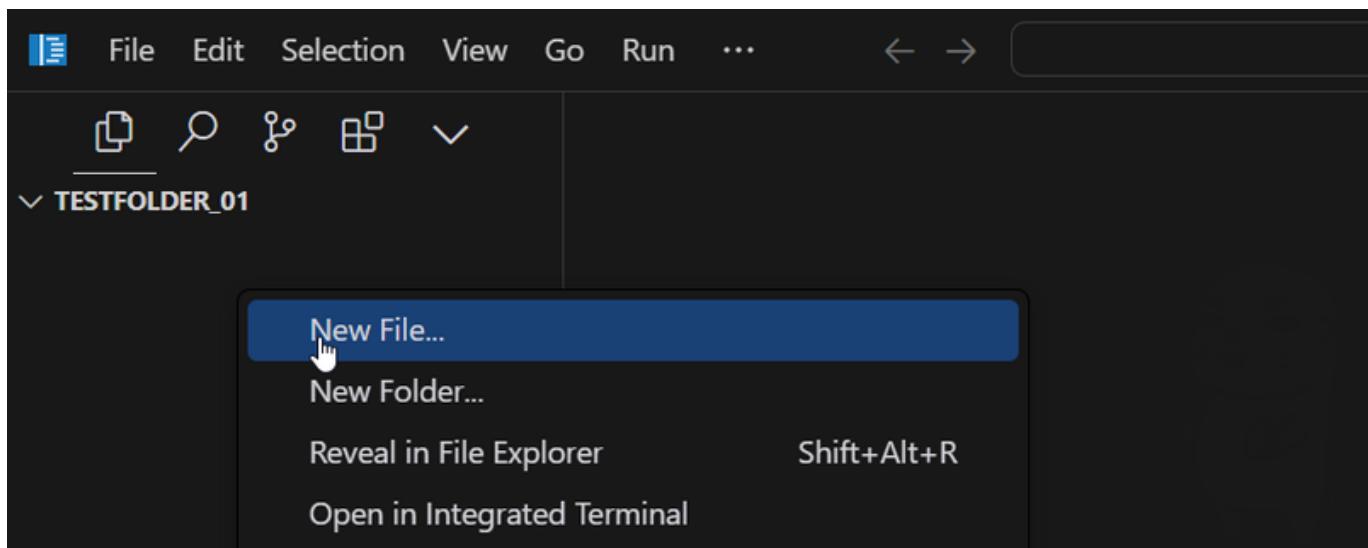
Whoooaaa! It's writing a whole bunch of stuff!!!

### Save and Run the Program

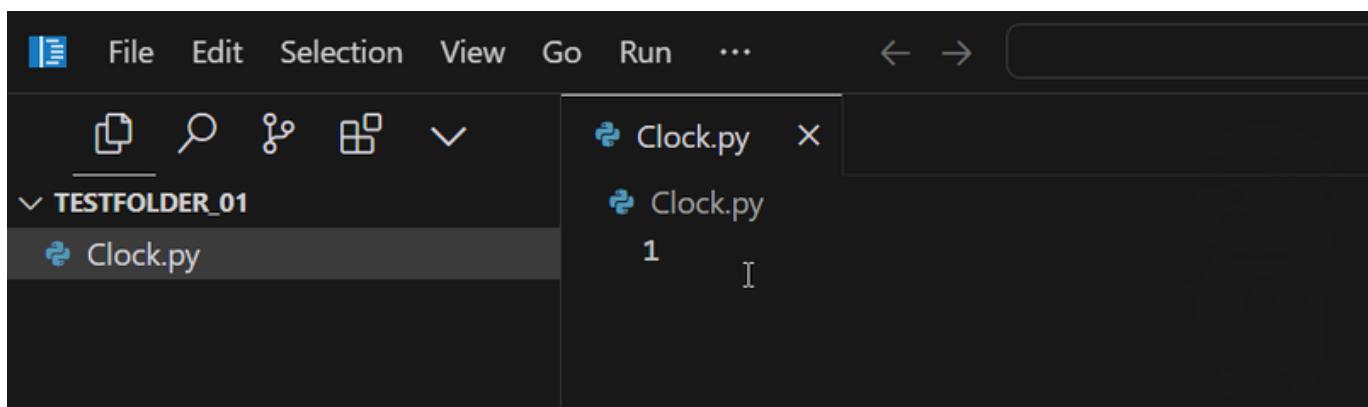
Once cursor finishes writing the program, let's save it as a file! Create an empty file and paste the program into it!

#### ♦ Creating an Empty File

Right-click on the left panel (file/folder view) and select **New File**.



Name the file **Clock.py**.





.py is the file extension, right?

Do I have to write it myself?

.py is the extension for Python script files.

Yes, you need to write the extension yourself! You got this!

## ◆ Saving the Program

Copy the program into Clock.py.

In the chat box, hover over the section with the program, and a **Copy button** will

appear. Click it to copy the program.

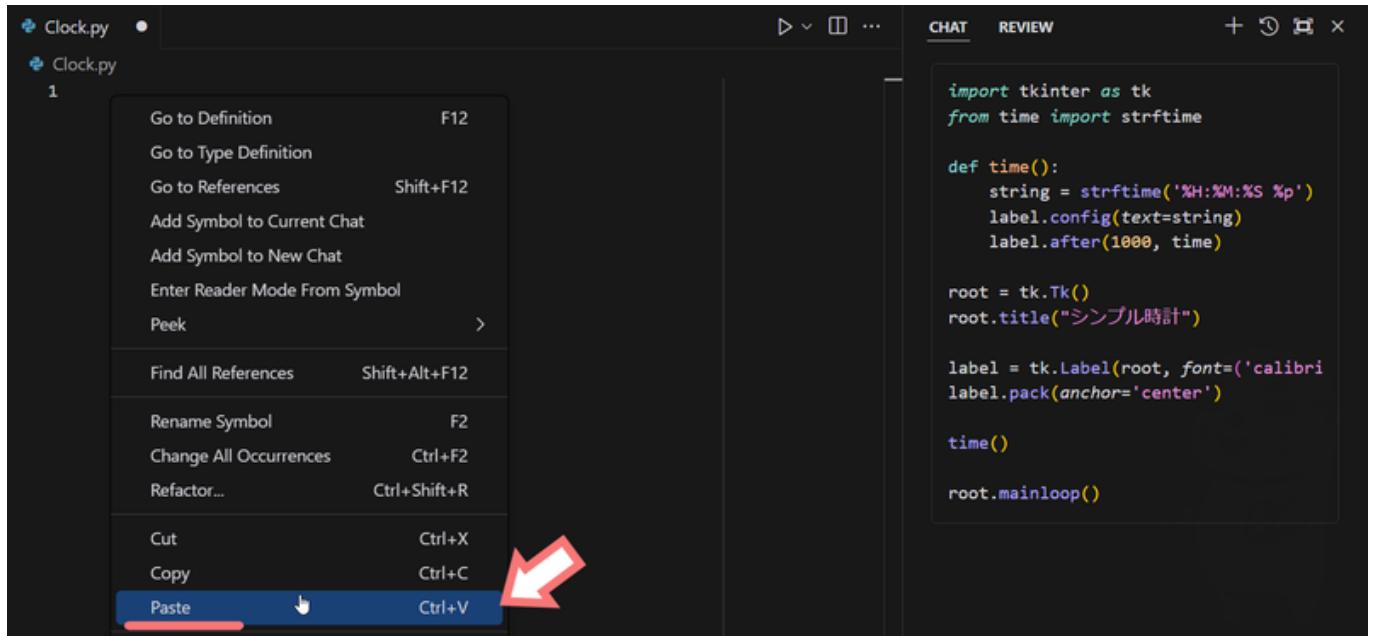
```
python
import tkinter as tk
from time import strftime

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

Then, paste the program into the central code-writing area.

Right-click -> Paste.

## Chapter 2: The First Challenge!



A screenshot of a code editor window titled "Clock.py". A context menu is open over some code, with the "Paste" option highlighted. A red arrow points from the "Paste" option towards the bottom right of the image.

```
import tkinter as tk
from time import strftime

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

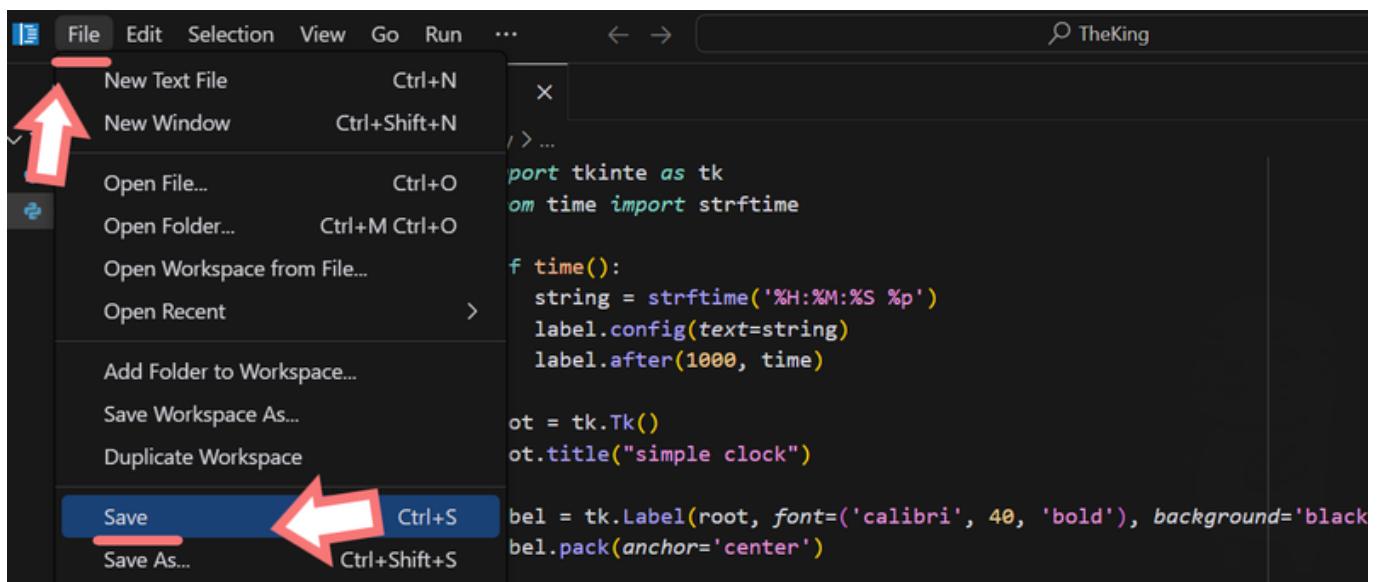
root = tk.Tk()
root.title("シンプル時計")

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

time()

root.mainloop()
```

After pasting the program, go to File -> Save.



A screenshot of a code editor window showing the "File" menu. The "Save" option is highlighted with a red arrow. Another red arrow points from the "Save" option towards the bottom left of the image.

```
port tkinter as tk
om time import strftime

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

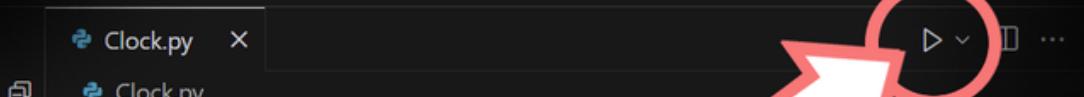
ot = tk.Tk()
ot.title("simple clock")

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

## ◆ Running the Program

Finally! It's time to run it!

Press cursor's Run button ► to execute the program!



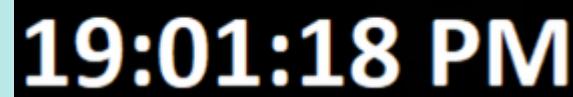
```
import tkinter as tk
from time import strftime

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

if __name__ == "__main__":
    label = tk.Label(root, font='arial 180', bg='black', fg='white', width=12, height=2)
    label.pack()
    update_time()
    root.mainloop()
```



## Clickin' the run button...



Huh...



Wowwww!!  
It's done!!!!  
Amazing!!

Did the clock window appear for everyone?

## If Things Don't Go Well



Wahhh!

It's getting all jumbled and confusing!

### ♦ Try Restarting Cursor

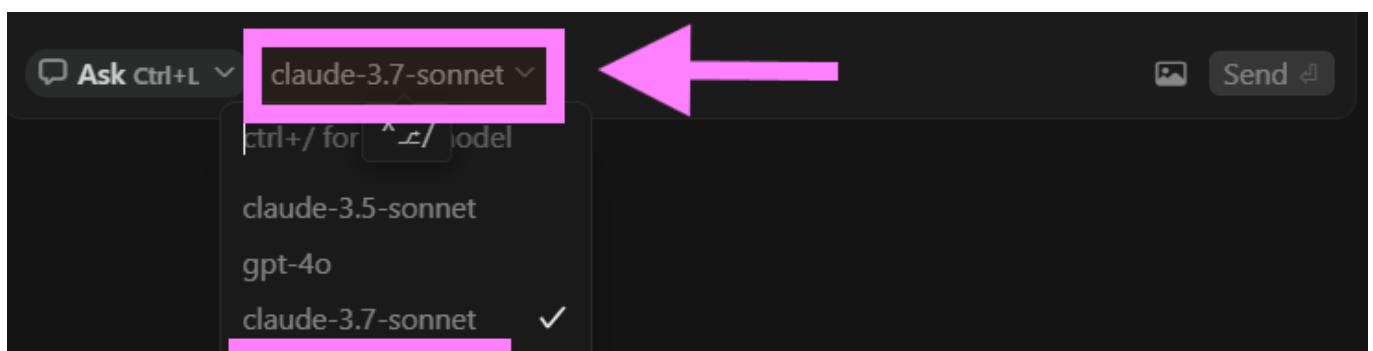
If Cursor isn't working properly, try closing and restarting it. This can solve many issues.

### ♦ Ask Cursor for Help

Cursor can help with more than just programming. If you're unsure about something, feel free to ask! It'll surely give you some useful hints.

### ♦ Try Changing the Model

If the responses are weird and you're stuck, select the chat box and choose a different model from the model selection menu. This time, try selecting **claude-3.7-sonnet**.



## Let's Customize the Program

Next, let's customize this clock!



I want the color to be orange.  
Getting particular about it.

Now, let's send an additional request! If you have more requests, there's a chat box in the bottom right corner!

```

File Edit Selection View ... ← → ⌂ TheKing
THEKING Clock copy.py Clock.py ...
1 import tkinter as tk
2 from time import strftime
3 | Ctrl+L to chat, Ctrl+K to generate
4 def time():
5     string = strftime('%H:%M:%S %p')
6     label.config(text=string)
7     label.after(1000, time)
8
9 root = tk.Tk()
10 root.title("simple clock")
11
12 label = tk.Label(root, font=('calibri', 40,
13 label.pack(anchor='center')
14
15 time()
16
17 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=('calib
label.pack(anchor='center')

time()

root.mainloop()

```

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

gpt-4 chat ctrl+D codebase

### ◆ How to Write a Request

@Clock.py

Change the color! I want it to be orange!

v gpt-xx v

Send ↴

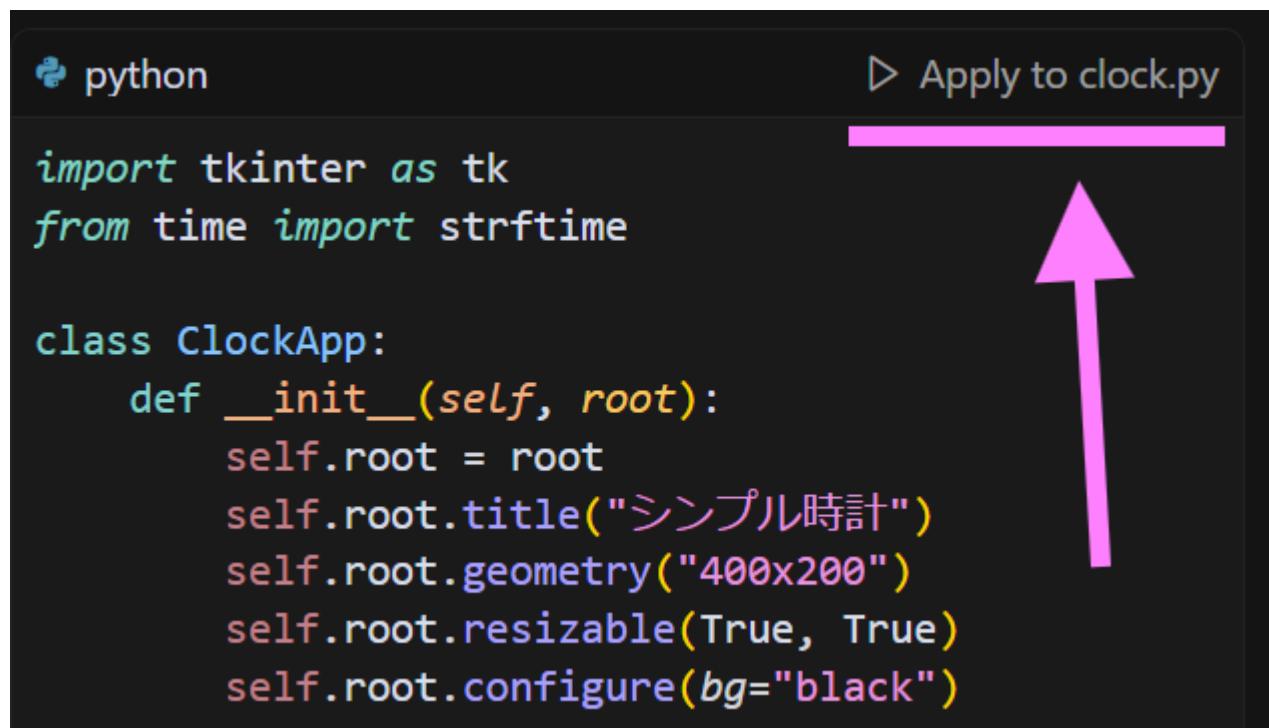
Press Enter and let cursor write the program for you!

### ◆ Automatically Apply the Fix

Once cursor writes the fix, let's apply it to the editor.

No need to copy-paste everything by hand!

Move your mouse cursor close to the program part in the reply, and when you see the **Apply button**, click it, and cursor will automatically apply the code.

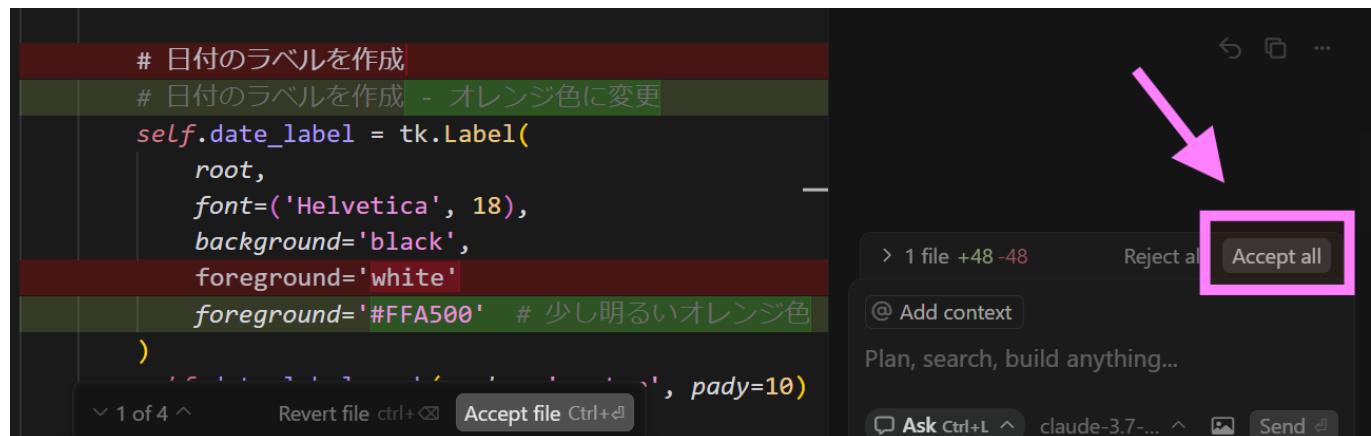


```
python
▶ Apply to clock.py
import tkinter as tk
from time import strftime

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")
```

When you press the button, cursor will start updating the program.

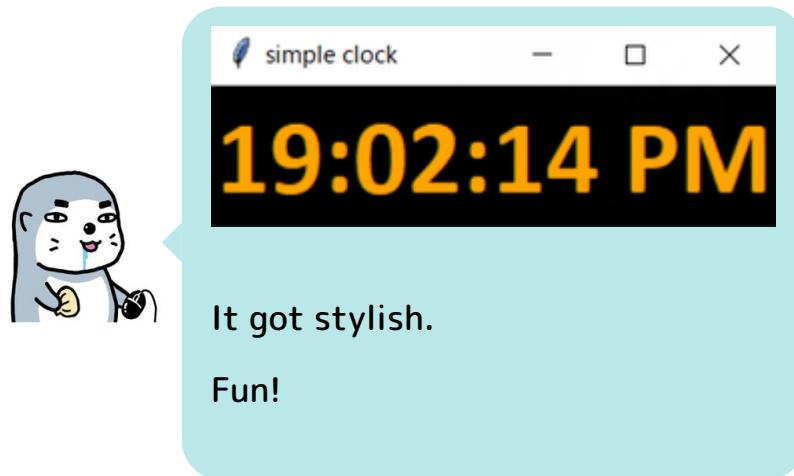
You should see the central screen being rewritten.



```
# 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'
)
self.date_label.pack(pady=10)
```

**The green part is cursor's fix suggestion.** At this point, it's just a suggestion, and the fix hasn't been applied yet, so you need to press **Accept** to confirm. Until you press Accept, the changes won't be reflected!!

Finally, press **Run ►** to check the result.

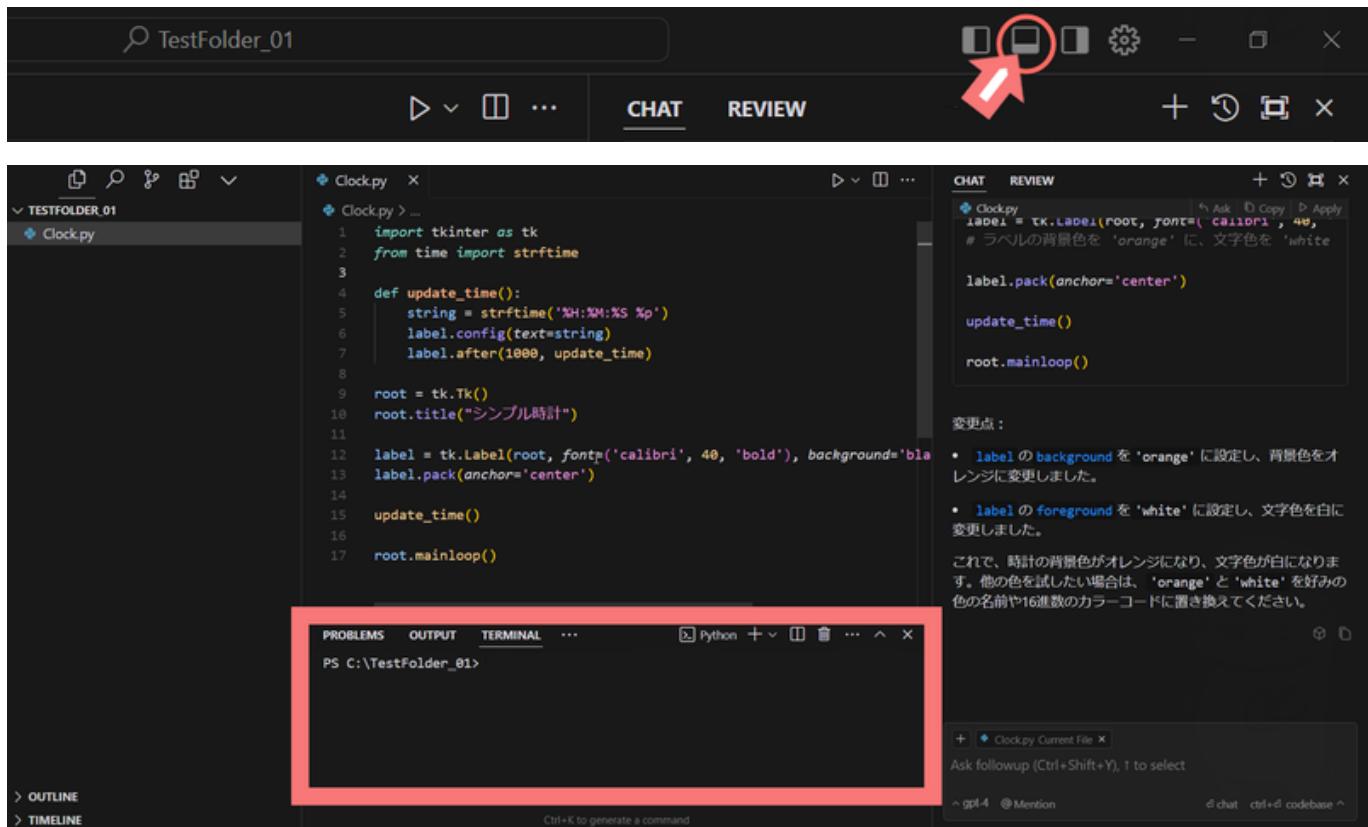


Just like this, you'll keep interacting with cursor until the tool matches your vision.

## Let's Handle Errors

If your program doesn't run, fix the errors while chatting with cursor.

① Press this button to open the Error Display Screen (Terminal).



② As a test, try rewriting "tkinter" to "tkinte" or something similar, and intentionally make a mistake and run it!

The screenshot shows the same code editor interface as before, but with a spelling error in the code. The first line now reads '1 import tkinte as tk'. The rest of the code is identical to the previous screenshot.

If the clock doesn't show up after running it, an error has occurred!

The Error Display Screen (Terminal) should be filled with text.

```
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 If you see the word "Error", drag and select the error description (if you're unsure, select everything).

④ Press "Add to Chat" to send the selected content to the chat box.



⑤ Mention that an error occurred, hit enter, and ask for a fix!

@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'
```

This kind of error appeared!

v gpt-xx v  Send ↴

Cursor will send you a fix, so use **Apply** to apply it to the editor.

## Chapter 2: The First Challenge!



Huh? Mine says "✓Continue" though...?

You're being asked if it's okay to apply this to the file you're currently working on. So, press ✓Continue to proceed.



I'm gonna continue!

After that, press Accept.

Once the green suggestion disappears, press Run ►!

Is the error gone?



The clock reappeared! Phew!

If you get stuck at any point, try repeating the steps while asking cursor for help.

## Launch the tool from the desktop



I wanna launch the clock we made from the desktop!

Sounds good! Let's ask cursor for help!

### ♦ How to ask

@Clock.py

I want to be able to launch my clock, Clock.py, from the desktop! What should I do? Please explain it briefly for beginners!

Ask claude-3.7-sonnet v

Send

### ♦ cursor's reply

The easiest way is to call the Python script (.py) from a batch file (.bat).

You can do this by following these steps:

1. Create a new text file on your desktop and rename it to "Clock.bat".
2. Edit this batch file and write the following content:

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

3. Replace the part C:\path\to\your\ with the actual path where your Clock.py file is located.

## Chapter 2: The First Challenge!

Now, you can just double-click the clock.bat on your desktop to run the Python script.



The response is completely different...

cursor's replies are different every time.

The example here is just that—an example. The reply you see on your screen might be different.

For now, let's try the "how to make a bat file" method.



I don't think the folder is "C:\path\to\your\Clock.py"...

I don't even know where the file is...

If you want to find the file location:

Right-click on the file name on the left screen -> Copy Path

Or right-click -> Reveal in Explorer



Alright! I'll rewrite it... There!



It launched when I double-clicked!

## Remember This!

If you're stuck, think back to this!

### ◆ Copy & Paste Shortcuts

「Ctrl」 + 「C」 to copy

「Ctrl」 + 「V」 to paste

If right-clicking doesn't work, use this!

'Course I know that much.



### ◆ How to Insert Line Breaks

- If you want to insert a line break in the chat, instead of pressing "Enter,"

「Shift」 + 「Enter」

Huh...



# Chapter 3: Let's Create Handy Tools

---

Automate a lot of tasks to make things easier

## What You'll Create

- File List Creation Tool
- Tool to Check Excel Lists

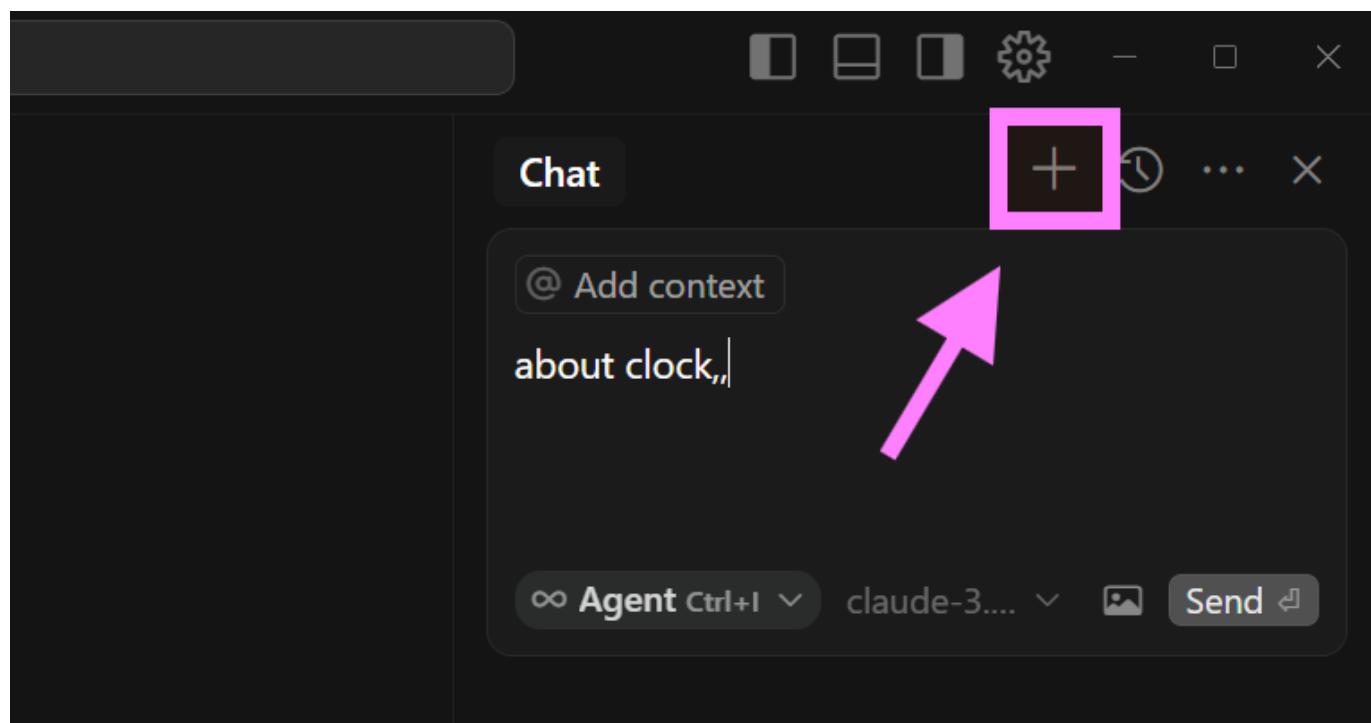
I wanna make things easier no matter the cost.



## Let's Start from New Chat

Before sending a new request, let's **reset the chat box**.

Try pressing the + button below.



Did the chat box reset?

If you keep chatting with the cursor for a long time, it tries to answer based on the initial info you entered, which can confuse the cursor. So, make sure to **press it often**.

## Let's Make a File List Tool



Boss wants me to make a file list...

Even though there's 1000 files...

Let's ask cursor for help!

If there's a specific format, it's a good idea to include a sample!

### ♦ How to Make a Request

@

Make a tool that creates a file list from a folder!

I need a list with the folder name, file names, and file sizes!

Something like this!

#Example:

C:\Osama\Images\

--- Image001.png(100KB)

--- Image002.png(12KB)

Ask claude-3.7-sonnet v

Send ↗

### ♦ cursor's Response

Let's create a Python script that generates a file list from a folder:

```

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})")

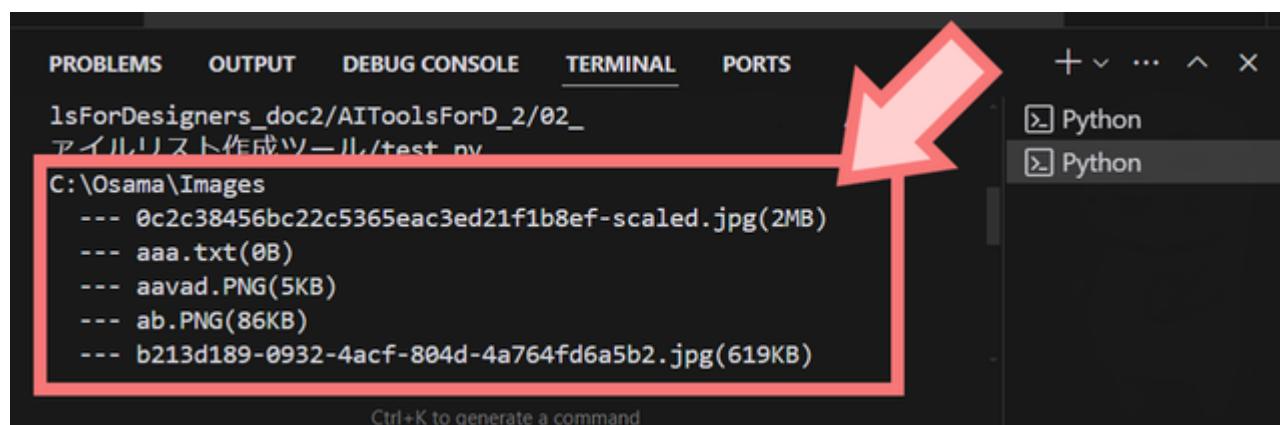
# Example usage

folder_path = r"C:\Osama\Images"
list_files(folder_path)

```

## ◆ Let's Try Running It

Let's see the results right away! Create a file called `list_test.py`, paste the program, and press the **Run Button ▶**. Then, check the terminal screen below!



```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS + ⌂ ⌄ ⌁ ⌃ X
lsForDesigners_doc2/AIToolsForD_2/02_
ファ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)

Ctrl+K to generate a command

```



I want it saved as a text file.

### ◆ Various Customizations

Let's ask for some changes. Keep requesting and applying the responses.

@ ❁list\_test.py

Output the result to a text file!

Ask claude-3.7-sonnet v

Send ↴

@ ❁list\_test.py

Add a window UI too!

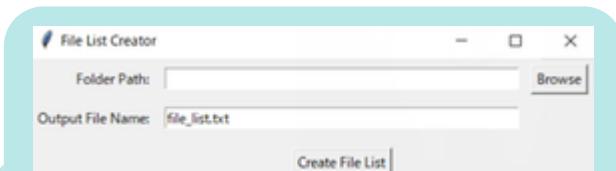
Ask claude-3.7-sonnet v

Send ↴

### ◆ Completion



1000 files done in an instant...!



## Let's Make It Stylish



I wanna make the UI cuter~

Alright, let's send a request to make it cuter!

Here, we'll use a library called **PySide6**.



Huh... what's PySide6?

Don't worry about it for now, just send the request like this!

### ◆ How to Write the Request

@ ✧list\_test.py

Make this script stylish with dreamy pastel colors using PySide6

Ask claude-3.7-sonnet v

Send ↗

### ◆ cursor's Response

Sure, let's use PySide6 to change the GUI to a stylish dreamy pastel color scheme. Here's the updated code:

```
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
.
.
.
(truncated)
```



Okay, copy-paste and run!

Wait, what? There's an error?

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



What does that even mean?

Let's ask cursor.

## ◆ How to Write the Request

@

It says "ModuleNotFoundError: No module named 'PySide6'"

What should I do?

I'm a beginner, so please explain it simply and briefly.

v gpt-xx v

submit ↴ ctrl + ↴ codebase

## ◆ cursor's Response

This error indicates that PySide6 is not installed. To fix it:

1. Open Command Prompt.
2. Run the following command:

```
pip install PySide6
```

This should install PySide6 and resolve the error.



pip is a tool for adding new features to Python.

It lets you easily install special features or extensions (libraries) that you normally can't use, and add them to your program.

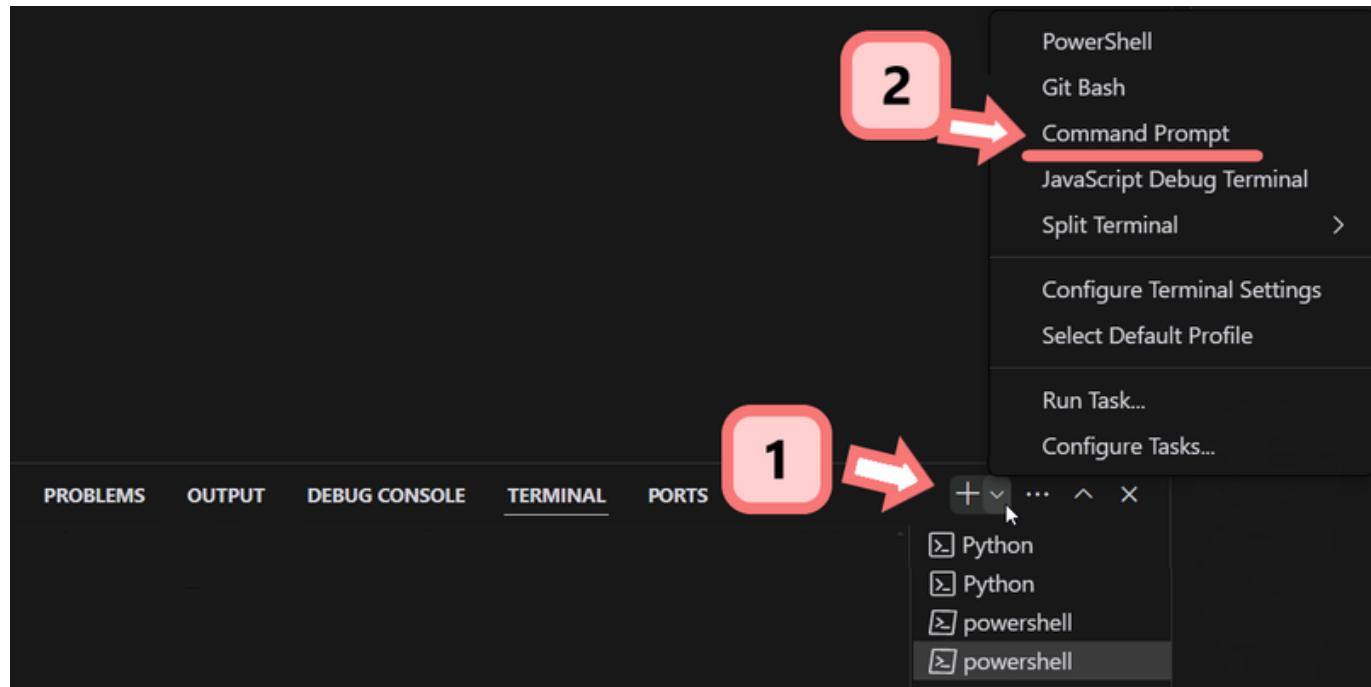
If you want a cute UI, you gotta install it!



Pain in the butt  
but gotta make it cute, so oh well.

### ◆ Entering Commands in the Terminal

Open the **Command Prompt** in the terminal screen.



Then, enter the command in the terminal to install it!

```
pip install PySide6
```

Copy this with **ctrl + C**, and paste it in the terminal with **ctrl + V**!

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

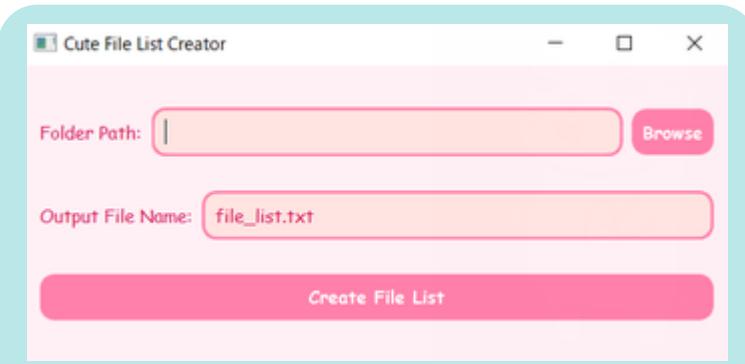
C:\test\test>pip install PySide6
```

Press **Enter** to start the installation.



Once the installation is done, try running the Python file you just created again by clicking **Run ▶**.

## ◆ Done



Super cute!

PySide6 rocks!

## Let's Make an Excel Check Tool



Boss is being unreasonable again!  
Telling me to check the Excel list against the filenames!  
There's like 1000 files!

Alright, let's get the data out of Excel!

### ◆ Install pip

First, let's install the necessary packages using pip!

```
pip install pandas openpyxl
```



I'm gonna shove it in the terminal!

### ◆ How to Ask for Help

@

I want to compare the list in Excel (C:\TheFile.xlsx, column name is  
FileName) with the filenames in the folder C:\Osama.

I need to check if the files in the Excel list are actually in the folder!

Is there a good way to do this?

Ask claude-3.7-sonnet v

Send ↗

## ♦ Response from cursor

Yes, there's a good way to do this using Python. The following script reads the list of filenames from the Excel file and compares it with the files in the specified folder.

```
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("Files in the Excel list but not in the folder:")
for file in missing_files:
    print(f"- {file}")
print("\nFiles in the folder but not in the Excel list:")
for file in extra_files:
    print(f"- {file}")
print(f"\nTotal: {len(missing_files)} files are missing.")
print(f"Total: {len(extra_files)} extra files are in the folder.")
```

Before running this script, you'll need to install the following libraries:

`pip install pandas openpyxl`



Yeah, yeah.

Copy-paste it into "exceltest.py" and run it!



The answer's in the terminal! Phew!

### ♦ Further Customization

@ExcelCheck.py

I want to be able to specify the Excel file!

Ask claude-3.7-sonnet v

Send ↗

@ExcelCheck.py

Make a UI! Use PySide6 and make it super cute!

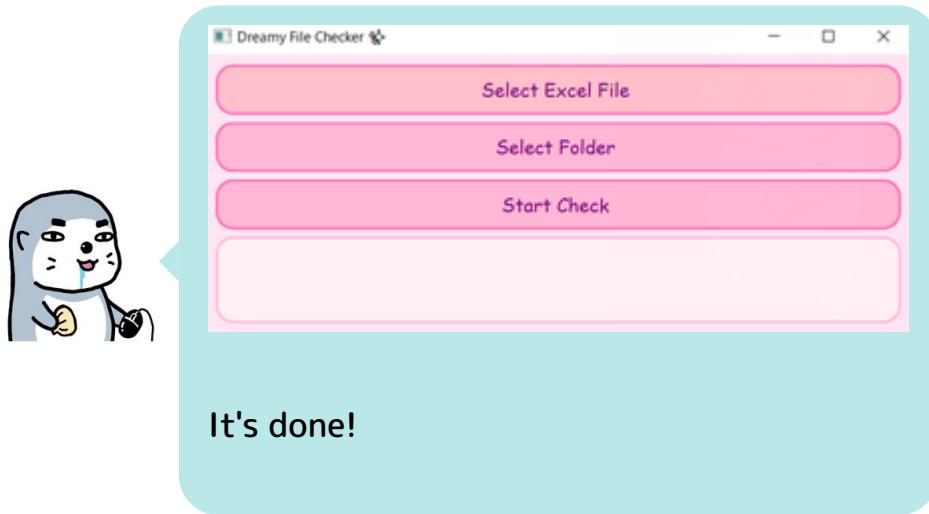
Ask claude-3.7-sonnet v

Send ↗



I learned PySide6!

### ♦ The Result



## You can make tools like these

You can easily create tools like these too.

- Make a tool that converts Excel files to XX format!
- Make a color picker!
- Make a to-do tool!
- Make a file backup tool!
- Turn the tool you made into an .exe!

Feel free to try asking for different things.

So many dreams!



# Chapter 4: Let's Absorb Knowledge

---

Let's have them answer after absorbing knowledge

What will be completed

- Presentation materials

Isn't it a tool?



## First

### ♦ Unique Languages for Each Software

Each software has its own "language." For example:

- Web browsers use HTML
- Excel uses VBA
- Maya uses MEL

This is similar to how different regions have different spoken languages.

### ♦ Handling Unfamiliar Languages

cursor has languages it's good at and languages it's not so good at. When you need it to write in a language it's not familiar with, you can use the **knowledge absorption feature** to help it learn and respond. This chapter explains how to use that feature.

The basic approach is the same for any software, so for now, just follow along with the steps as a foundation.

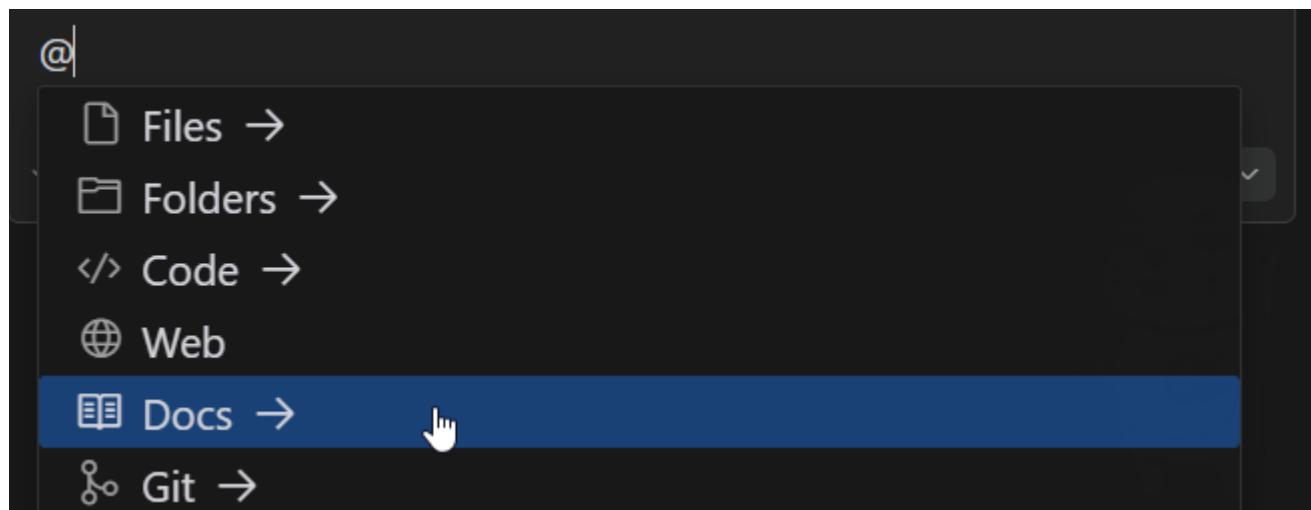
## Let's Create Presentation Materials with Marp



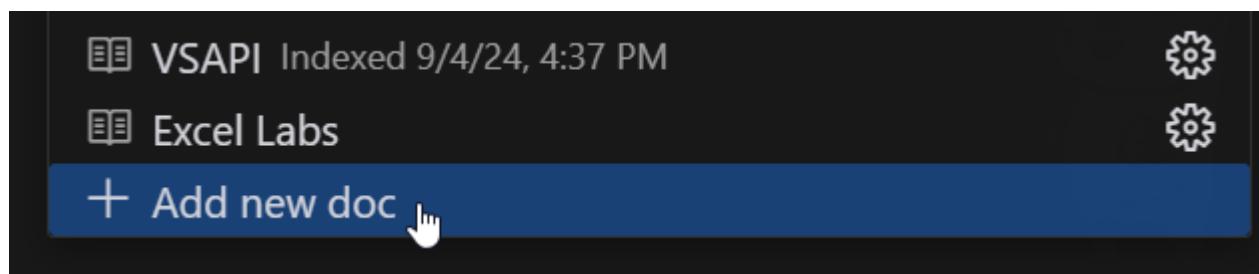
As a specialist in presentation formats, you'll be absorbing knowledge about "Marp" here.

### ◆ Absorb New Knowledge into cursor

- ① Type "@" in the chat or click @AddContext to display the list. Select "Docs".



- ② Select "+ Add New Doc".

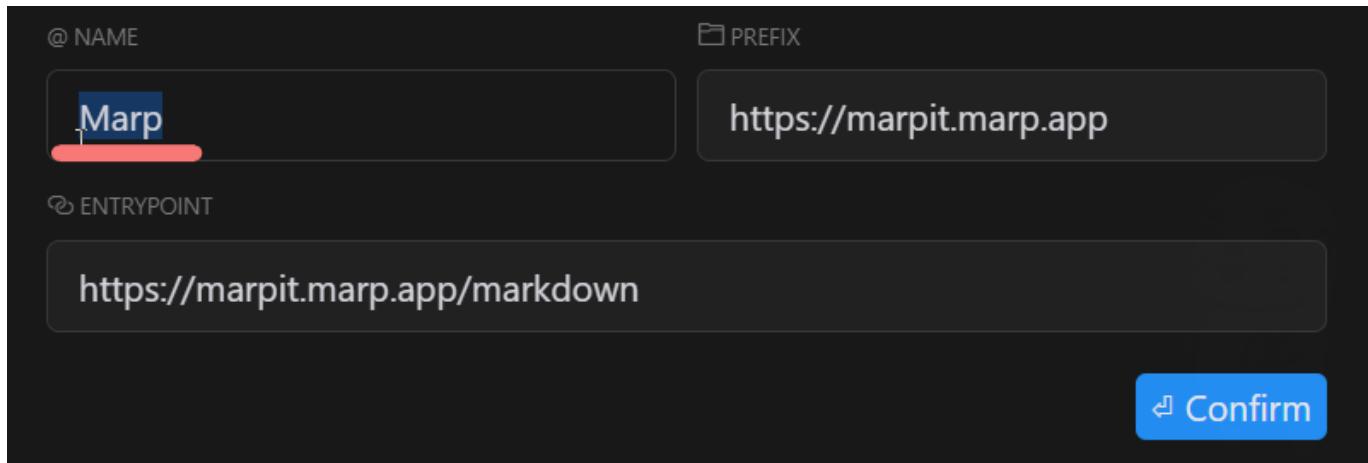


- ③ Enter the homepage address where Marp materials are published.

Enter this address: <https://marpit.marp.app/markdown>



④ Give it a name you like and **confirm**. Let's name it **Marp** here.



Well done. The cursor has absorbed the information.



Ugh, do I have to do this every time?

No need to do it every time.

Once the information is absorbed, you can call it up anytime by typing "@Marp".

## ◆ Making Requests Based on Absorbed Knowledge

- Start your request with the name you gave earlier, @Marp.
- (Feel free to rewrite the content in your own style)

@  Marp

I wanna do a presentation at work, so can you format it nicely in Marp style?

#Presentation Topic

Snacks are delicious! You know, snacks are tasty, right? Keeping some nice snacks in the fridge for emergencies means even if something bad happens, I can be like, "Hey, I've got snacks!" and stay strong. So, I wanna say it's a good idea! So, in conclusion, I want a nice fridge in the office.

v gpt-xx v

Send 

With this, the cursor will write a response with the absorbed knowledge!

---

marp: true

theme: default

---

Snacks are… (omitted)

Save the response as **MarpTest\_01.md**.

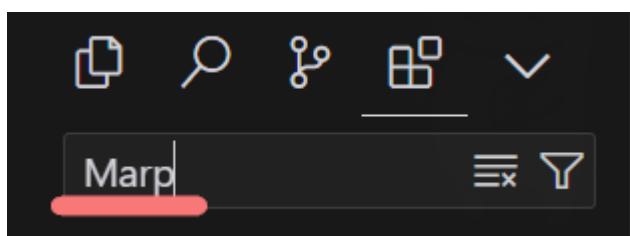
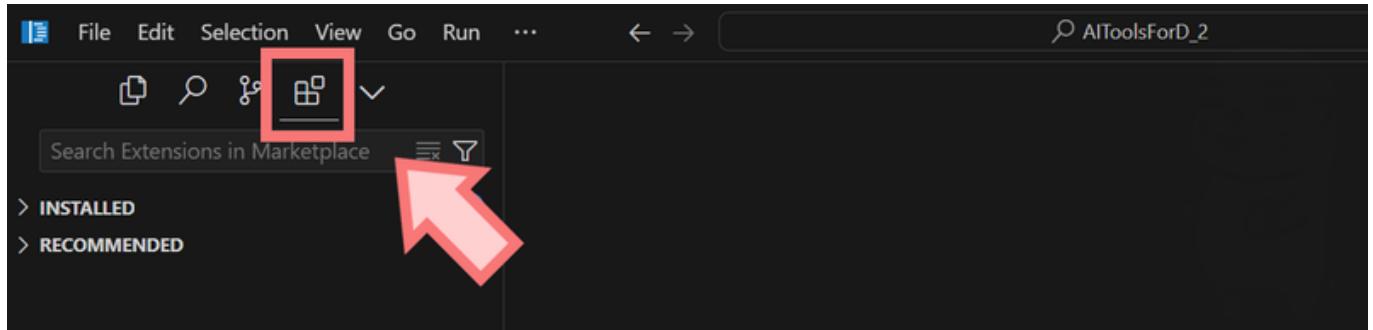


So it's not ".py" this time, huh?

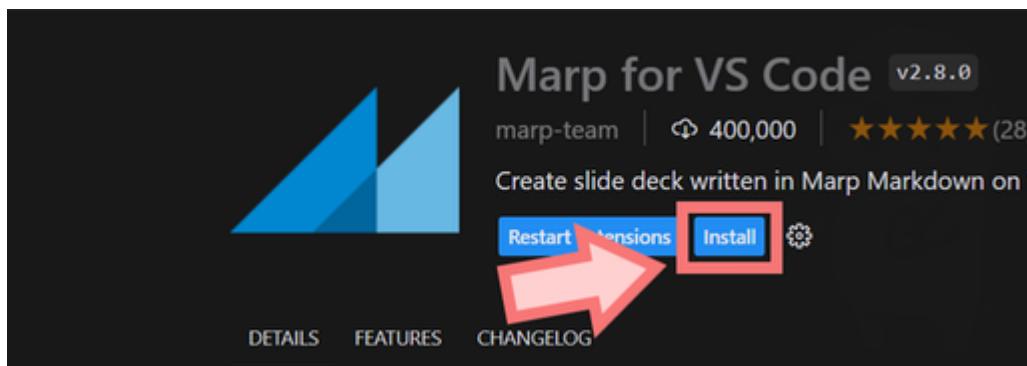
## ◆ Displaying in the Actual Format

Install Marp on the cursor.

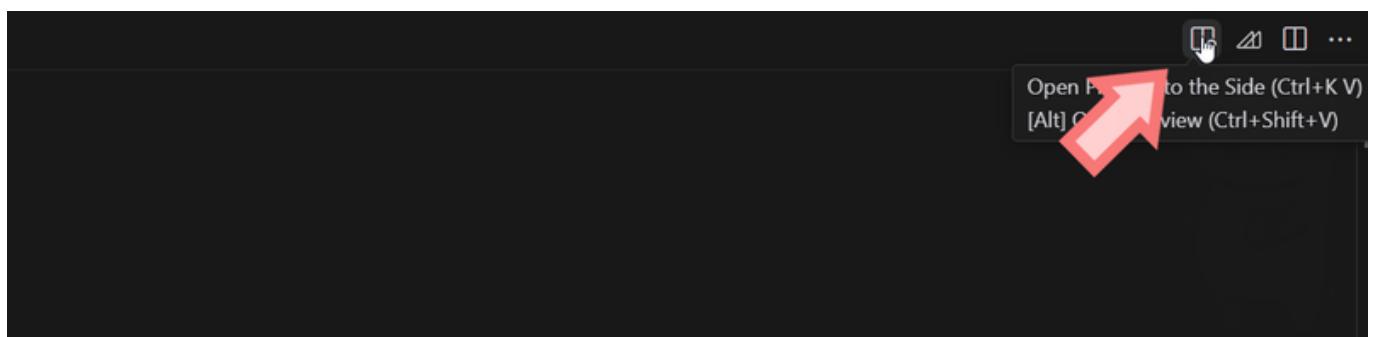
① Click the extension icon and search for "Marp".



② Once found, please install it.



③ After installation, click the button to display with Marp.



The presentation format is displayed on the right side!



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 > ...  
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 generated presentation slide. A red arrow points from the text "# おやつはおいしいという話" in the code to the corresponding slide content. The slide content is:

おやつはおいしいという話



Oh wow!

The presentation materials are done in no time!

## ♦ Customize the Look

- Let's make it dreamy and cute!

@ ▼MarpTest\_01.md 📖 Marp

Let's make it dreamy and cute!

v gpt-xx v

Send ↗

## ♦ Result

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!"



It's super cute!

## ♦ You Can Ask for Non-Program Requests Too



Yay! I'm gonna ask for a lot!!!

@ ▼MarpTest\_01.md 📖 Marp

Make it so my boss really wants to buy a fridge!!!

v gpt-xx v

Send ↗

## ♦ Result

### 💼 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

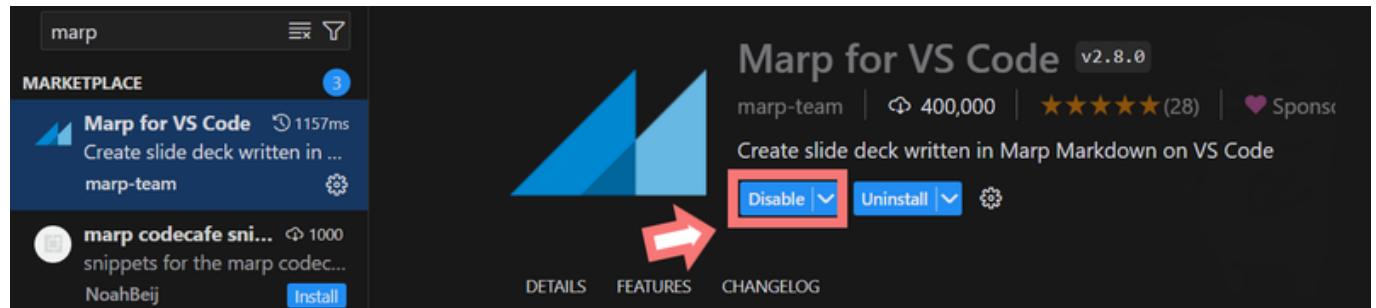


Yeah, that's exactly what I wanted to say.

By using @ to absorb knowledge, you can handle all sorts of specifications like this!

## ◆ Disable Marp When You're Done

When you're done, go to the extensions screen, select Marp, and click "Disable" to turn it off.



### Various Ways to Absorb Knowledge

#### ♦ Personal Ranking

##### First Place: Docs

You know the Docs feature. Just input a URL, and it will fetch everything under that hierarchy.

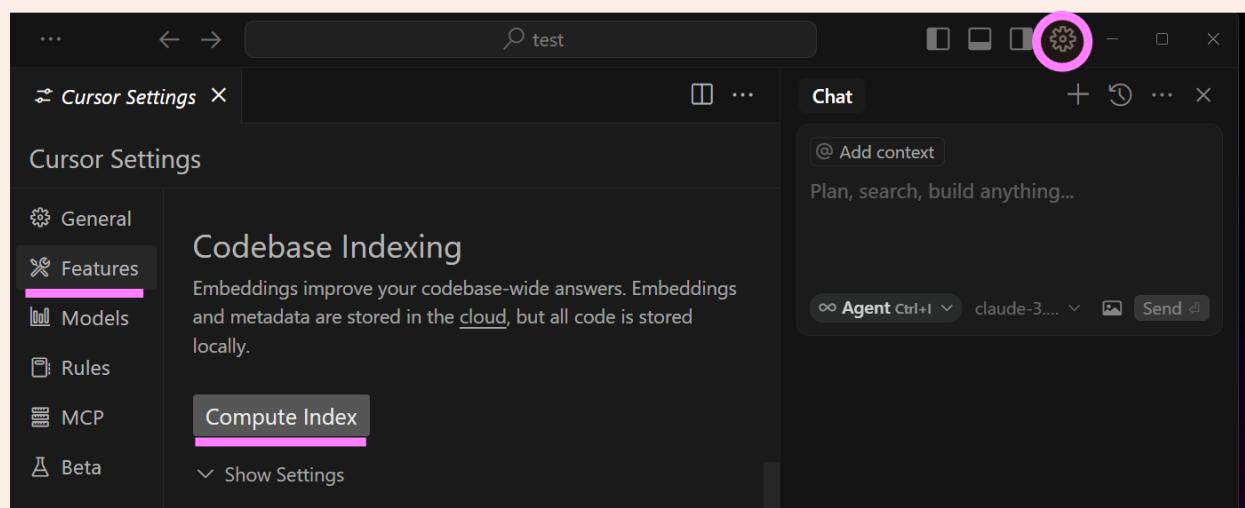
##### Second Place: Codebase:

It references the entire hierarchy under the open folder.

Since the Docs feature has a limit of 1000 pages, if you want to reference more than that, save the files locally and use this. Also, if Docs can't read something properly, you can save the webpage or PDF as text and use Codebase.

You need to do some prep work with CodeBaseIndexing to use it.

Settings button ⚙ -> Features -> CodebaseIndexing



By pressing this, you can store the contents under the hierarchy into a kind of knowledge bag called Codebase. After that, you can pull knowledge from the bag by using @codebase. If the folder contents change, make sure to update it.

If you want to improve accuracy when using @codebase, setting Reasoning Step to "yes" will boost its search power.

### Third Place: <https://~>

Paste a URL, and it will reference that webpage. This is handy when you only want it to look at a single page.

## ♦ Features I Use Occasionally

**Git:** A feature for those using Git. If you choose @commit and ask it to write a commit message, it will generate one for you.

**Files:** A feature to specify and load individual files. Since I often just attach single files via the "+" button, I don't use this much.

**Web:** It searches the entire web, and cursor autonomously gathers information. It answers based on the collected data. The results are hit or miss, so I haven't been able to use it effectively.

## Don't Expect Too Much



Alright! Gonna make it absorb a massive amount of info!

Wait a second!

LLMs can't process large amounts of text all at once, so they search for relevant parts to answer from. They don't fully understand everything!

Also, when LLMs are overloaded with information, their reasoning ability drops. So, feeding them tons of knowledge doesn't necessarily lead to better answers. In fact, giving them less knowledge tends to improve accuracy.



What a pain... So sad...