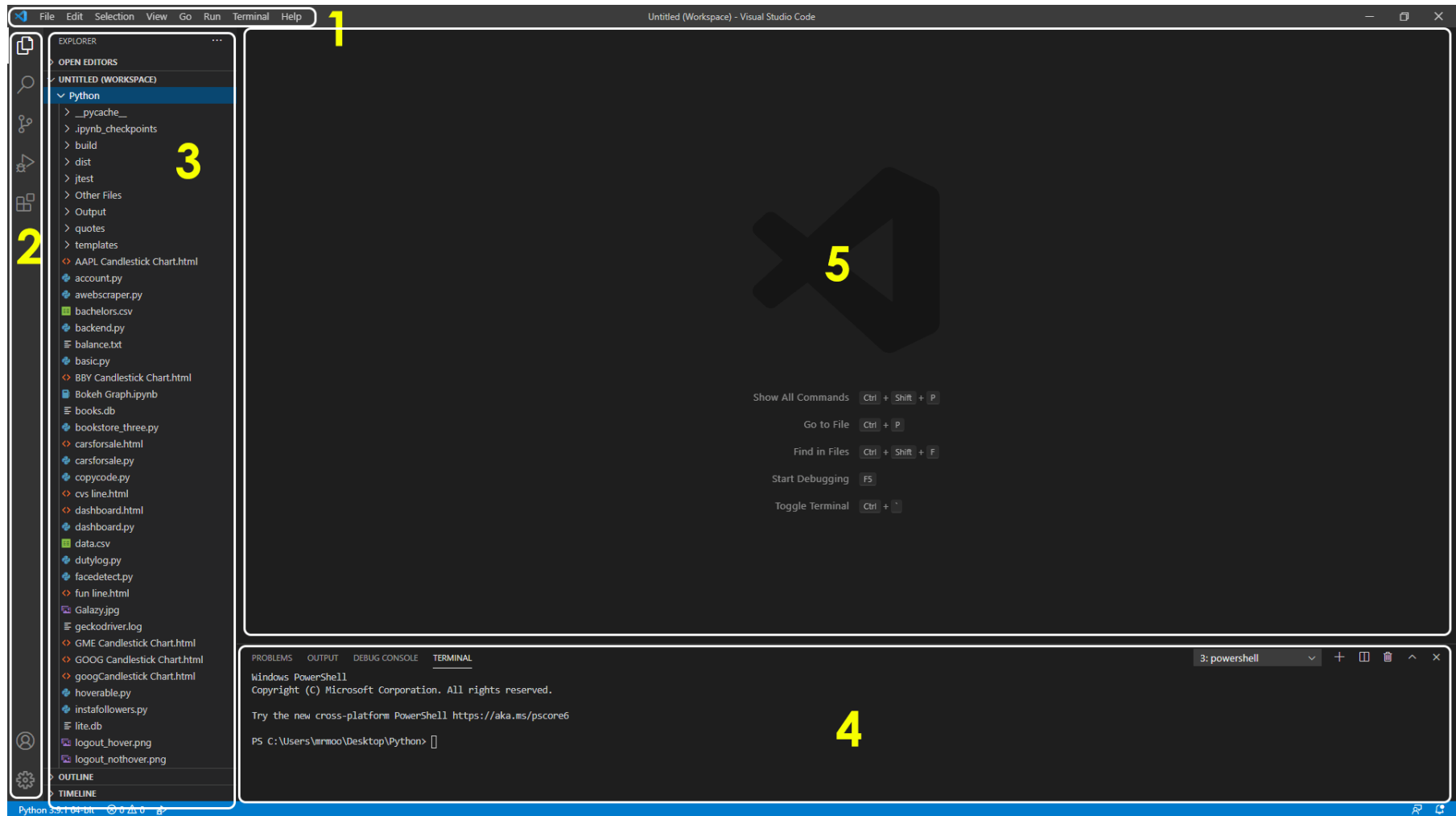


[Previous: VSCode Python Extensions](#)[Next: Python in VSCode](#) >[Home](#) » [Creating Python Programs](#) » VSCode GUI tour: Explore The Elements of This Powerful IDE

## VSCode GUI tour: Explore The Elements of This Powerful IDE

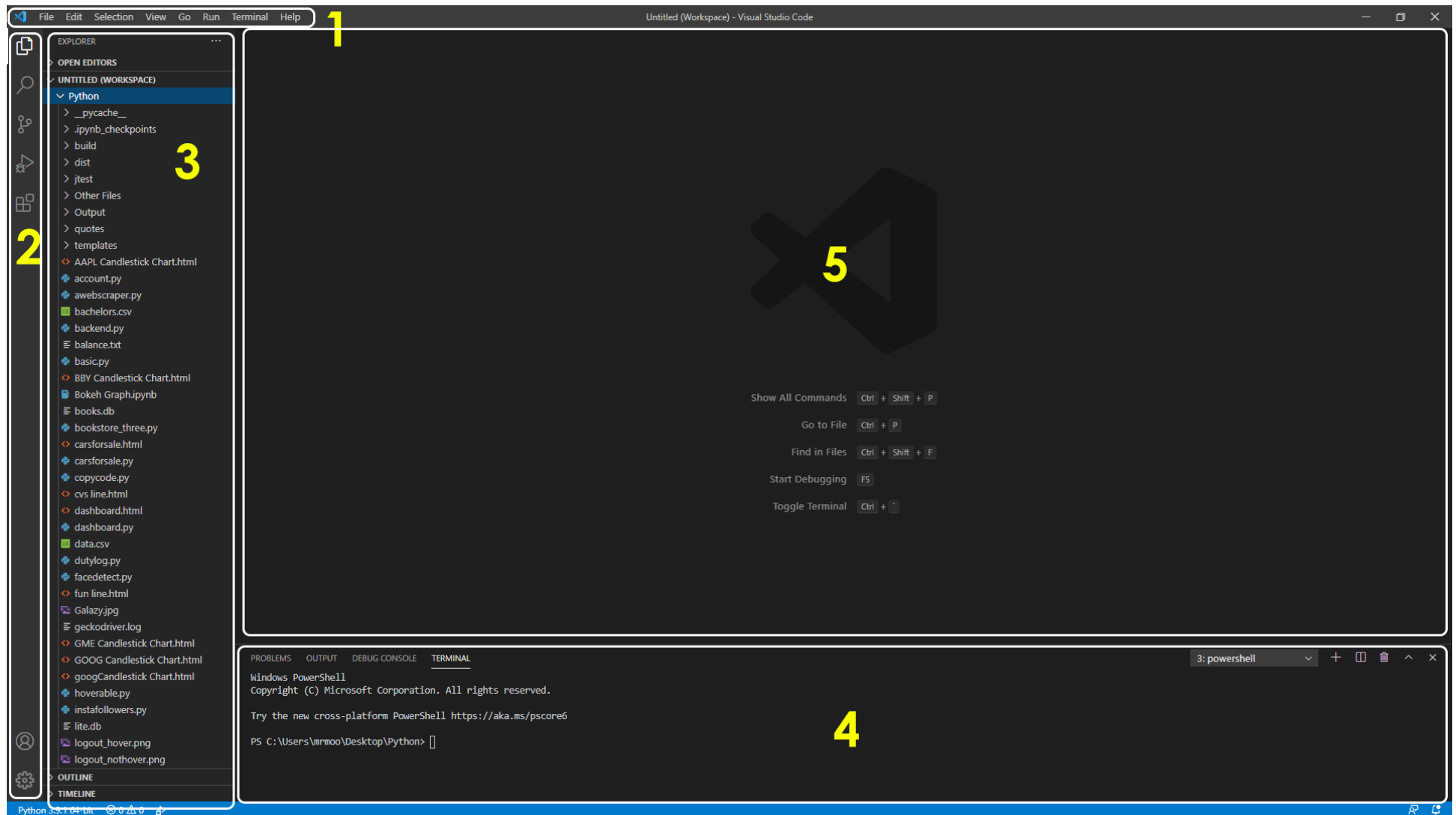
September 20, 2022

It might look overwhelming when you fire up VSCode for the first time. Let me walk you through the GUI so you get the most from this powerful IDE.

# Main sections of the VSCode workspace

The primary workspace in VSCode is divided into five main sections, as noted by the numbers:

1. The **menu bar**
2. VSCode **Sidebar**
3. VSCode **Explorer**
4. Debug **Console/Terminal**
5. **Document** Display/**Editor**



VSCode overview

Let's start by looking at these sections in detail!

## The menu bar

# File

New File	Ctrl+N
New Window	Ctrl+Shift+N
Open File...	Ctrl+O
Open Folder...	Ctrl+K Ctrl+O
Open Workspace...	
Open Recent	>
Add Folder to Workspace...	
Save Workspace As...	
Save	Ctrl+S
Save As...	Ctrl+Shift+S
Save All	Ctrl+K S
Auto Save	
Preferences	>
Revert File	
Close Editor	Ctrl+F4
Close Folder	Ctrl+K F
Close Window	Ctrl+W
Exit	

# Edit

Undo	Ctrl+Z
Redo	Ctrl+Y
Cut	Ctrl+X
Copy	Ctrl+C
Paste	Ctrl+V
Find	Ctrl+F
Replace	Ctrl+H
Find in Files	Ctrl+Shift+F
Replace in Files	Ctrl+Shift+H
Toggle Line Comment	Ctrl+/ /
Toggle Block Comment	Shift+Alt+A
Emmet: Expand Abbreviation	Tab

# Selection

Select All	Ctrl+A
Expand Selection	Shift+Alt+RightArrow
Shrink Selection	Shift+Alt+LeftArrow
Copy Line Up	Shift+Alt+UpArrow
Copy Line Down	Shift+Alt+DownArrow
Move Line Up	Alt+UpArrow
Move Line Down	Alt+DownArrow
Duplicate Selection	
Add Cursor Above	Ctrl+Alt+UpArrow
Add Cursor Below	Ctrl+Alt+DownArrow
Add Cursors to Line Ends	Shift+Alt+I
Add Next Occurrence	Ctrl+D
Add Previous Occurrence	
Select All Occurrences	Ctrl+Shift+L
Switch to Ctrl+Click for Multi-Cursor	
Column Selection Mode	

# View

Command Palette	Ctrl+Shift+P
Open View...	
Appearance	>
Editor Layout	>
Explorer	Ctrl+Shift+E
Search	Ctrl+Shift+F
SCM	Ctrl+Shift+G
Run	Ctrl+Shift+D
Extensions	Ctrl+Shift+X
Output	Ctrl+Shift+U
Debug Console	Ctrl+Shift+Y
Terminal	Ctrl+`
Problems	Ctrl+Shift+M
Toggle Word Wrap	Alt+Z
✓ Show Minimap	
✓ Show Breadcrumbs	
✓ Render Whitespace	
Render Control Characters	

Back	Alt+LeftArrow
Forward	Alt+RightArrow
Last Edit Location	Ctrl+K Ctrl+Q
Switch Editor	>
Switch Group	>
Go to File...	Ctrl+P
Go to Symbol in Workspace...	Ctrl+T
Go to Symbol in Editor...	Ctrl+Shift+O
Go to Definition	F12
Go to Declaration	
Go to Type Definition	
Go to Implementations	Ctrl+F12
Go to References	Shift+F12
Go to Line/Column...	Ctrl+G
Go to Bracket	Ctrl+Shift+\
Next Problem	F8
Previous Problem	Shift+F8
Next Change	Alt+F3
Previous Change	Shift+Alt+F3

# Go

Start Debugging	F5
Run Without Debugging	Ctrl+F5
Stop Debugging	Shift+F5
Restart Debugging	Ctrl+Shift+F5
Open Configurations	
Add Configuration...	
Step Over	F10
Step Into	F11
Step Out	Shift+F11
Continue	F5
Toggle Breakpoint	F9
New Breakpoint	>
Enable All Breakpoints	
Disable All Breakpoints	
Remove All Breakpoints	
Install Additional Debuggers...	

# Run

New Terminal	Ctrl+Shift+`
Split Terminal	Ctrl+Shift+5
Run Task...	
Run Build Task...	Ctrl+Shift+B
Run Active File	
Run Selected Text	
Show Running Tasks...	
Restart Running Task...	
Terminate Task...	
Configure Tasks...	
Configure Default Build Task...	

# Terminal

Welcome	
Interactive Playground	
Documentation	
Release Notes	
Keyboard Shortcuts Reference	Ctrl+K Ctrl+R
Introductory Videos	
Tips and Tricks	
Join Us on Twitter	
Search Feature Requests	
Report Issue	
View License	
Privacy Statement	
Toggle Developer Tools	Ctrl+Shift+I
Open Process Explorer	
Check for Updates...	
About	

# Help

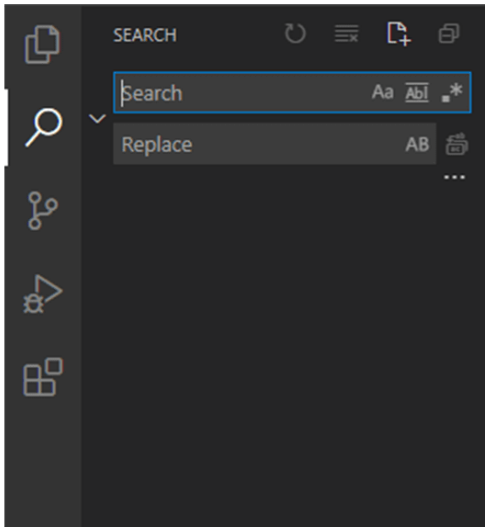
Menu's from VSCode's menubar

The menu bar is a no-frills part of the UI but vital to your projects.

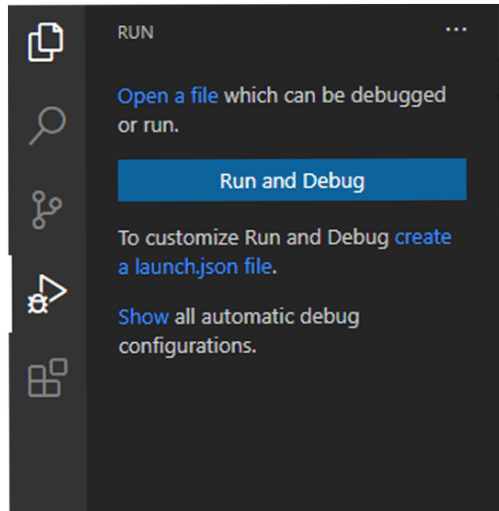
- Under the “**File**” section, you can **save and open projects**, but you can **add folders** to your workspace as well.
- The “**Edit**” tab contains the standard **copy/paste options**, which come in handy when you need to make copies of a line with slight modifications. The **find and replace** options are hidden here as well.
- Under “**Selection**,” there are options that allow you to **select and move through code easier**.
- The “**View**” tab carries options that will enable you to **change the look of VSCode** and show windows such as a **debug console**.
- I don’t use “**Go**” as much, but this section again allows you to **navigate code easier**.
- The “**Run**” tab offers options to **debug code and add configurations**.
- The “**Terminal**” window lets you create a Terminal window and edit tasks.
- Our toolbox’s last tool is the “**Help**” window, which leads you to more information and documentation about VSCode.

## The Sidebar

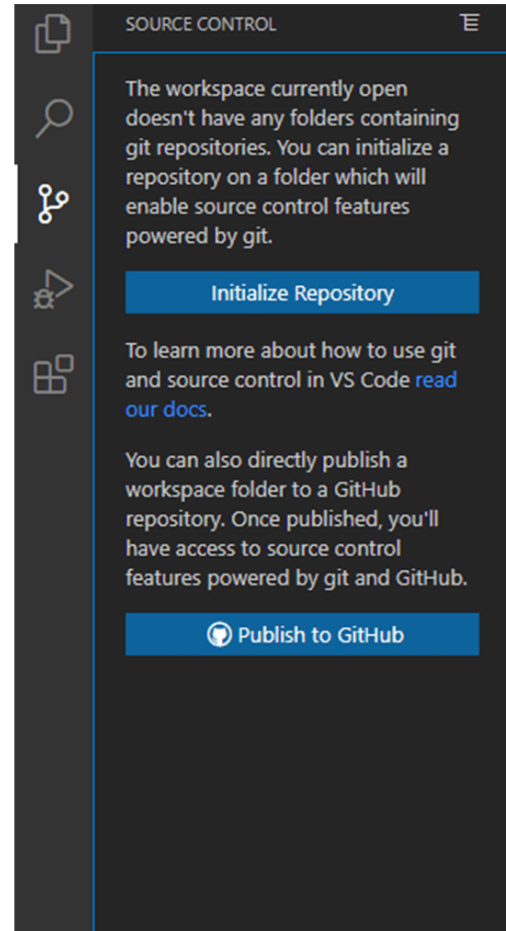
Looking closer at the sidebar, there are five main elements. The first is the “**Explorer**” tab, which I’ll go into in more detail later, but understand that it’s the meat and potatoes of organization in your workspace.



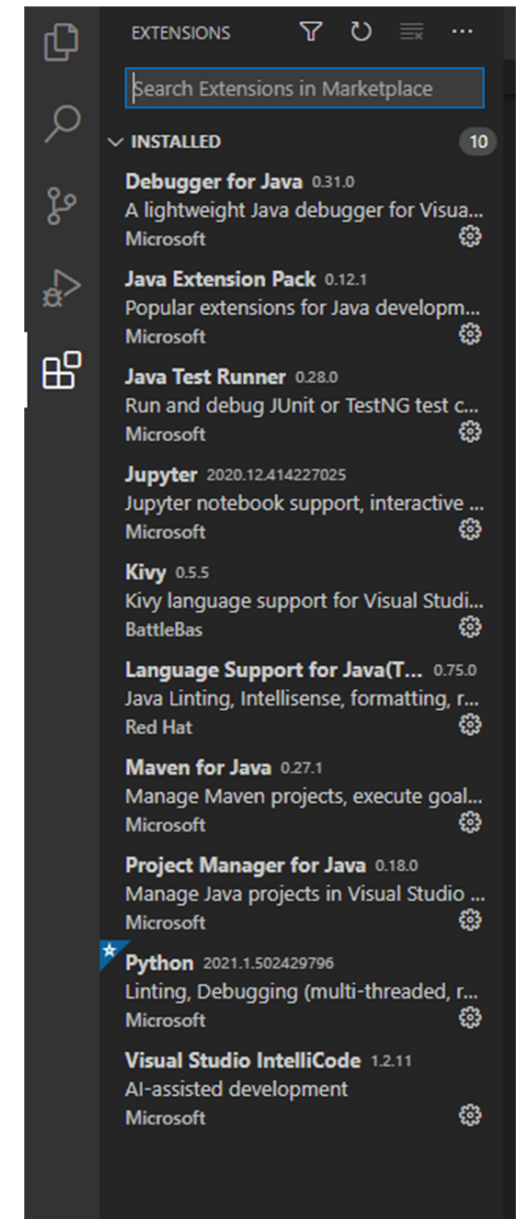
## Search



## Debug



## Sources



## Extensions

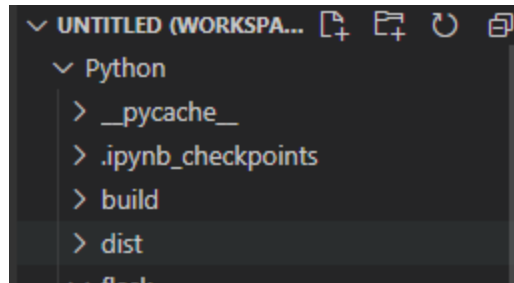
VSCode sidebar panels

The “**Search**” tab is **used to search** for any string of characters. The best part is that any file in your workspace will be searched too, meaning that you can find specific text in all of your source code.

The “**Source Control**” tab allows you to publish and maintain code on the GitHub repository. This allows your code to be made public for other programmers to use and build their projects from. While an excellent feature for more seasoned programmers, you won’t need this if you are still learning VSCode.

The last tab, “**Extensions**,” allows you to see currently installed extensions and find others available to add. As you can see, I have a few installed for the Java and Python programming languages. If you haven’t done so already, make sure you have the [essential VSCode extensions for Python](#) installed.

## Explorer

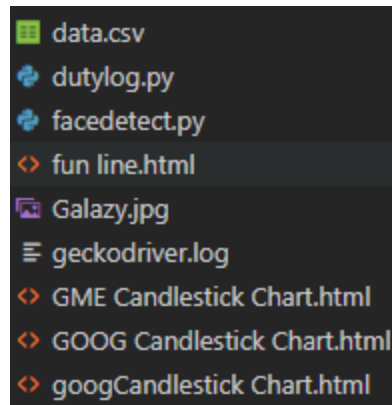


The explorer panel

The explorer lets you **create**, **organize**, and **delete** files. You can create a new folder or refresh the directory using the tool buttons at the top. Any changes to your directory are almost instantly shown here, but you might need to force a refresh now and then. Furthermore, the explorer panel allows you to view things like a document outline (handy for large [Markdown](#) files, for example).

The last **window icon** **allows** you to **minimize all folders at** once. You will have to manage several files simultaneously as you build larger projects such as websites. This button clears the clutter and folds everything back into a simple view.

Skipping the first icon was no accident, as I wanted to show a cool feature in VSCode. This little icon lets you create a file in your current directory.



You can create many files that contain source code and will be recognized by VSCode. For example, the `.csv` file extension is typically used for storing comma-separated data. The `.py` files contain Python code, while the `.html` files are webpage files. My favorite part of this feature is the little icons it creates based on your file type. It gives you visual cues to quickly find what you're looking for.

## Debug Console and Terminal Window

The small bottom window is typically reserved for a Terminal window. Integrating a terminal in your workspace saves time and clicking. If you want to navigate through other directories or run a script, that's typically done through the terminal, which is why it's nice to have a nested window at the bottom.

```
Requirement already satisfied: Jinja2>=2.10.1 in c:\python39\lib\site-packages (from flask) (2.11.2)
Requirement already satisfied: MarkupSafe>=0.23 in c:\python39\lib\site-packages (from Jinja2>=2.10.1->flask) (1.1.1)
PS C:\Users\mrmoo\Desktop\Python> python sandbox.py
Traceback (most recent call last):
  File "C:\Users\mrmoo\Desktop\Python\sandbox.py", line 16, in <module>
    name = section.find_all("span", {"class":"footable-toggle"})[0].text
  File "C:\Python39\lib\site-packages\bs4\element.py", line 2173, in __getattr__
    raise AttributeError(
AttributeError: ResultSet object has no attribute 'find_all'. You're probably treating a list of elements like a single element. Did you call find_all() when you meant to call find()?
PS C:\Users\mrmoo\Desktop\Python>
```

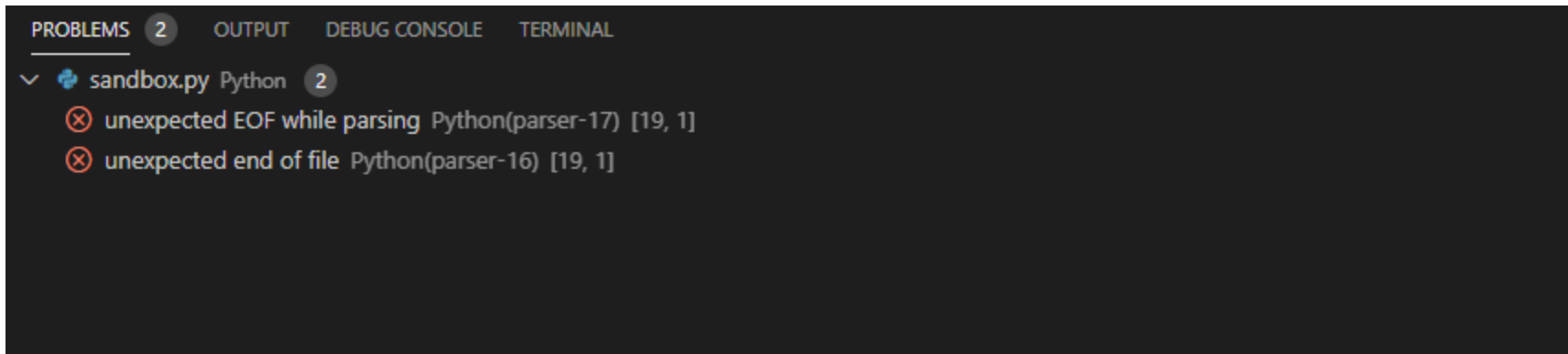
Running a script in VSCode is quite simple using the terminal. Let's say you have a script called `thisisavirus.py` saved in your directory. If you want to run that, simply type `python thisisavirus.py` in the console.

You might want to run a python file in a different directory than your workspace is in. In this case, point the terminal to the file name and still use the `python` command as seen below. It's important to use commas if the path has spaces in. This will prevent the error shown below from popping up.

```
C:\Users\Zach G>python "C:\Users\Zach G\Desktop\Python\carsforsale.py"
python: can't open file 'C:\Users\Zach G\Desktop\Python\carsforsale.py': [Errno 2] No such file or directory
```

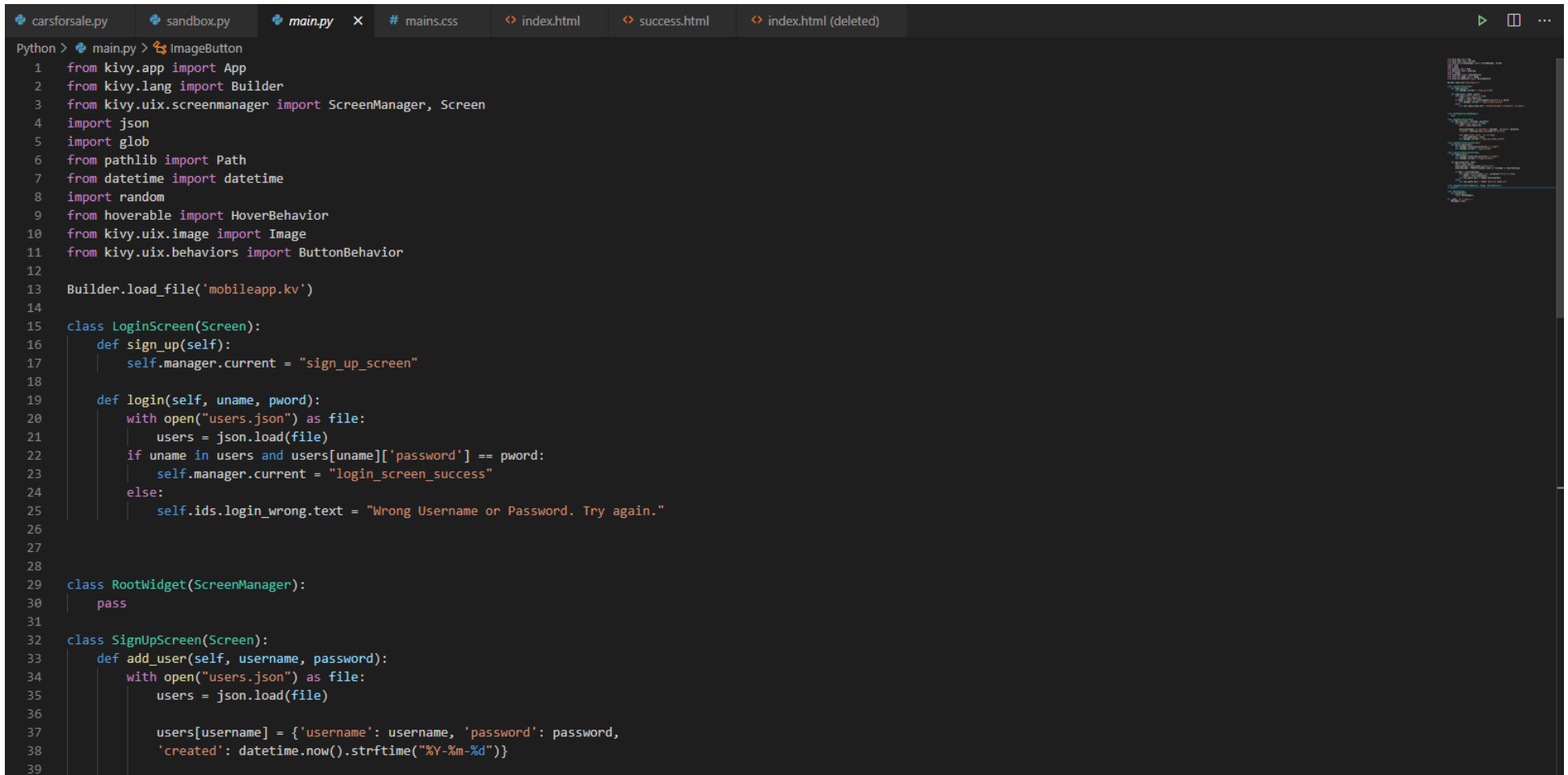
The terminal is also fantastic to see where errors in your code might be, such as shown above. The traceback shows you what went wrong and where the code is that gave the error.





My favorite of the tools in this section is simply called “Problems.” It checks every open file in your workspace for errors and other problems. As seen here, I have an “EOF” error, which is simply a syntax error because of missing parentheses in the command `print(name)`.

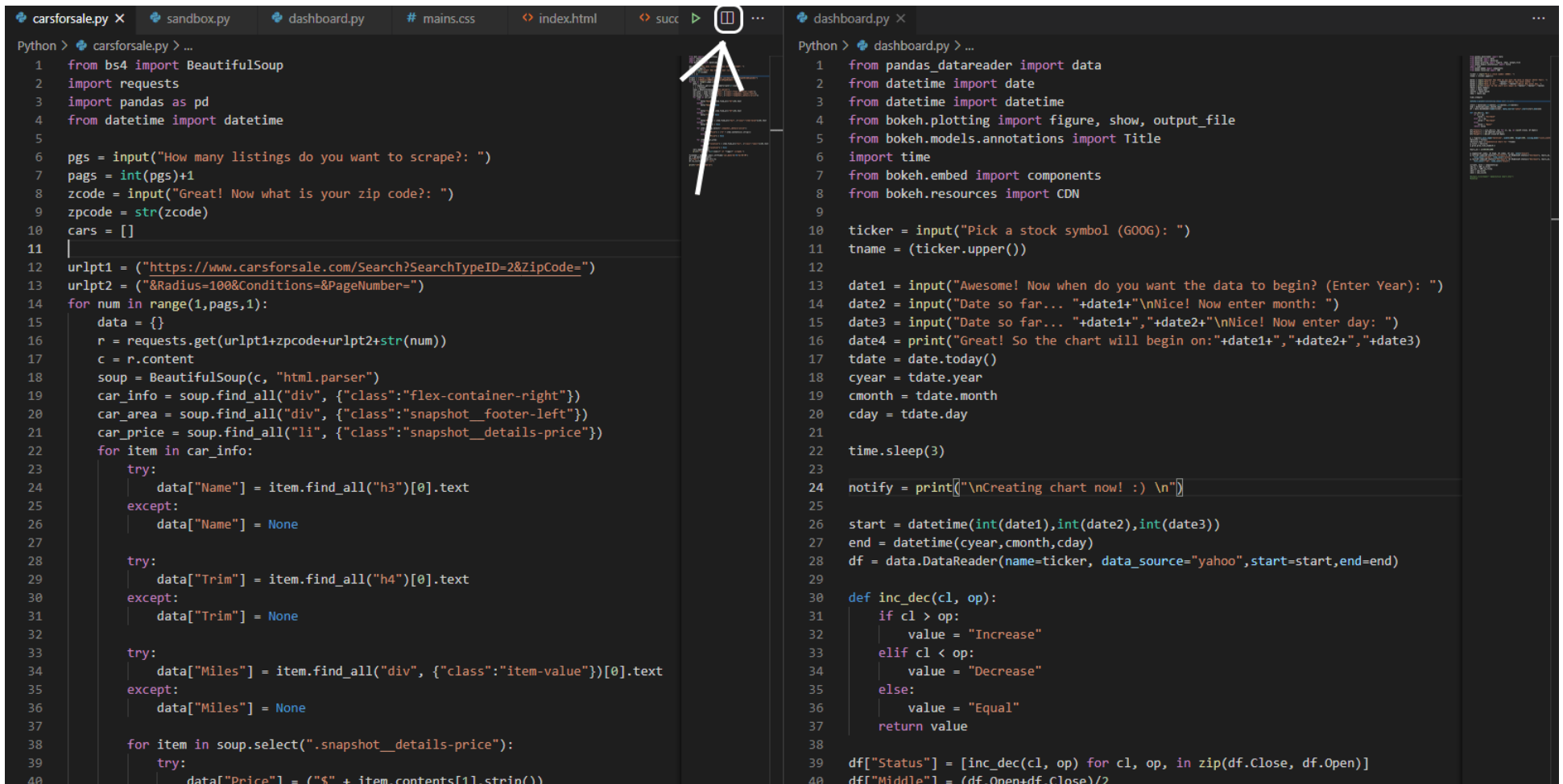
## The Document Editor



```
Python > main.py > ImageButton
1 from kivy.app import App
2 from kivy.lang import Builder
3 from kivy.uix.screenmanager import ScreenManager, Screen
4 import json
5 import glob
6 from pathlib import Path
7 from datetime import datetime
8 import random
9 from hoverable import HoverBehavior
10 from kivy.uix.image import Image
11 from kivy.uix.behaviors import ButtonBehavior
12
13 Builder.load_file('mobileapp.kv')
14
15 class LoginScreen(Screen):
16     def sign_up(self):
17         self.manager.current = "sign_up_screen"
18
19     def login(self, uname, pword):
20         with open("users.json") as file:
21             users = json.load(file)
22             if uname in users and users[uname]['password'] == pword:
23                 self.manager.current = "login_screen_success"
24             else:
25                 self.ids.login_wrong.text = "Wrong Username or Password. Try again."
26
27
28
29 class RootWidget(ScreenManager):
30     pass
31
32 class SignUpScreen(Screen):
33     def add_user(self, username, password):
34         with open("users.json") as file:
35             users = json.load(file)
36
37             users[username] = {'username': username, 'password': password,
38                               'created': datetime.now().strftime("%Y-%m-%d")}
39
```

Look at all those pretty colors! In this code I built for a class, there are about six different colors, each representing another type of code. For example, **purple** text highlights **native Python commands** such as “import” and words found in loops. **Strings** will be highlighted **orange** to stand out from other text as well.

Another **excellent feature of VSCode is the mini text overlay**. When working with many lines of code, an error will inevitably pop up. Rather than scroll through hundreds of lines, VSCode provides you with a smaller version of your code on the right side, which will **highlight a line in red if an error is spotted**.



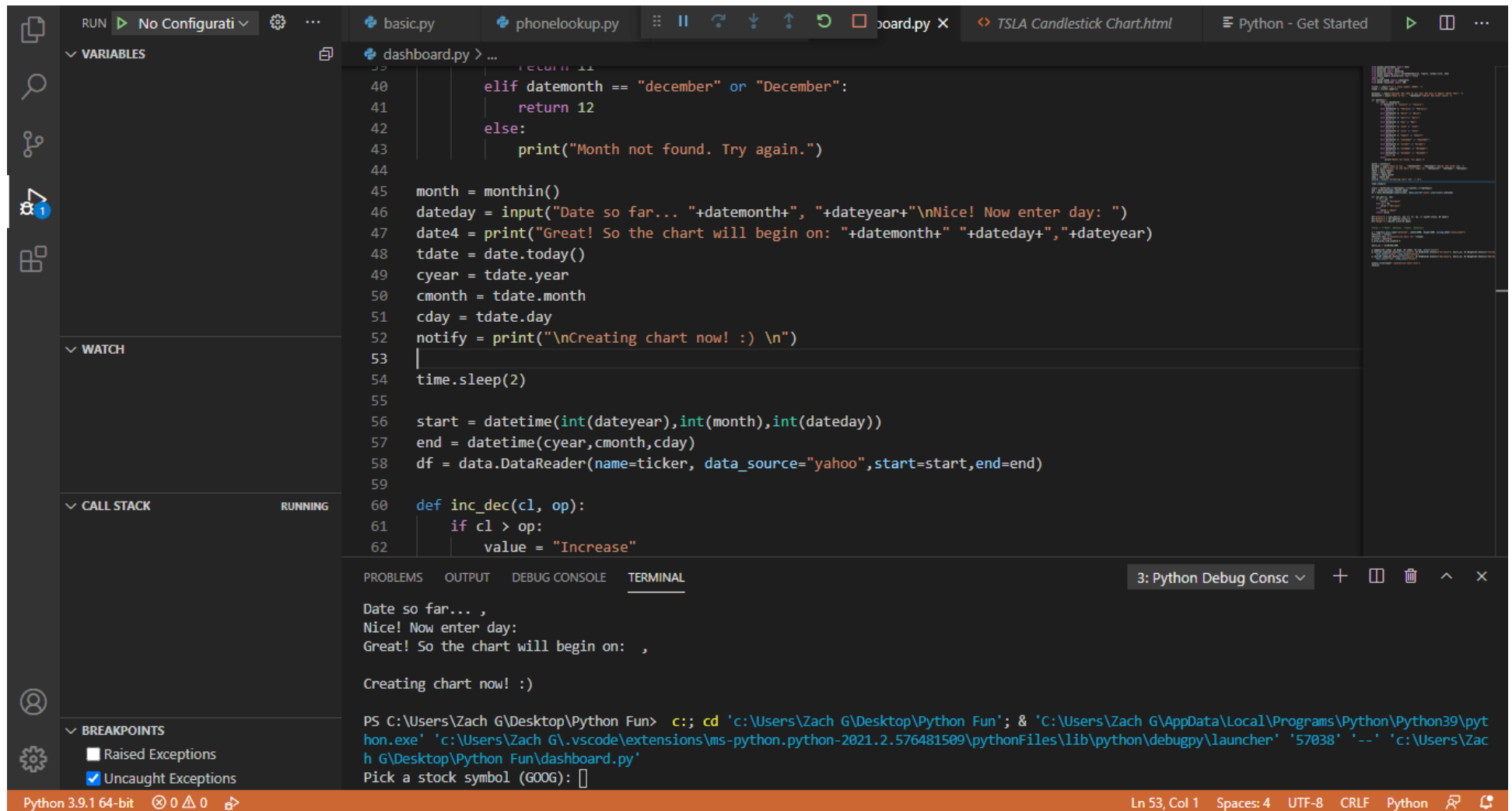
VSCode allows you to see multiple files by simply clicking on them and then clicking this little icon near the top. Each file is scrollable and dependent on the other file.

To the left of the dual-screen button is a tiny green arrow, which will run the code. More on this can be read in the next article on [running and debugging your code in VSCode](#).

Finally, think of those people with ten browser tabs open at once. You may despise their ways, but VSCode sets no limit of open tabs either. This does come in handy now, knowing that you can split the screen and edit multiple files at once.

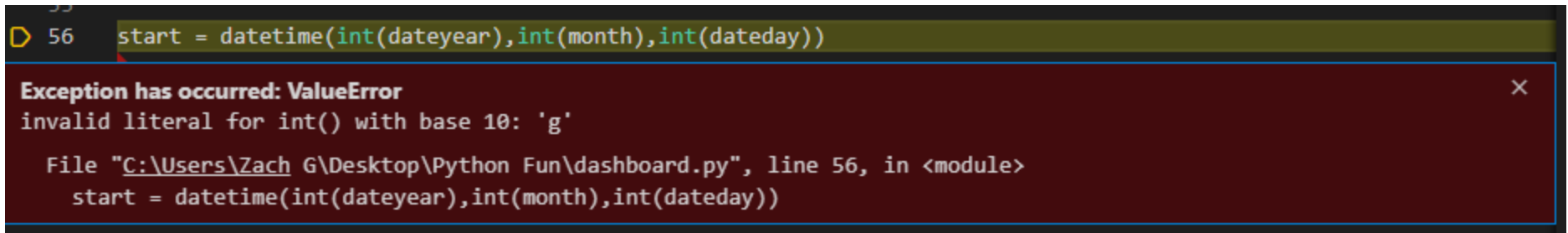
## Finding and squashing bugs

We haven't talked about the fourth icon yet. The **play button**, decorated with a little **bug**, runs your script in **debug mode**. When you enter **debugging mode**, the **bottom of VSCode will turn orange**. Your script is running in debug mode.



### Debugging software with VSCode

In **debug mode**, VSCode **checks each line for errors as it runs**. If a **bug is found**, you'll be **presented with this bright red window** that points you to where the error occurred. This makes finding errors and debugging code much easier than depending on the terminal to tell you which line was at fault.



VSCode highlighting an error in your code

Debug mode also allows you to step through your code. This means **running it line by line**, **stopping after each line**. This **gives you** the **opportunity to inspect** the **current state of variables and objects** as your code is running. It's a powerful way to squash those hard-to-find bugs and get a better understanding of what's happening.

For a more thorough dive into [running and debugging Python code in VSCode](#), just continue reading the next article.

Special thanks to [Zach Goreczny](#) for contributing to this article. Do you want to contribute to Python Land too? Please [contact us](#).

## Get certified with our courses

Learn Python properly through small, easy-to-digest lessons, progress tracking, quizzes to test your knowledge, and practice sessions. Each course will earn you a downloadable course certificate.

[Beginners Python Course \(2023\)](#)

[Modules, Packages, And Virtual Environments \(2023\)](#)

[NumPy Course: The Hands-on Introduction To NumPy \(2023\)](#)

[◀ Previous: VSCode Python Extensions](#)

[Next: Python in VSCode](#)