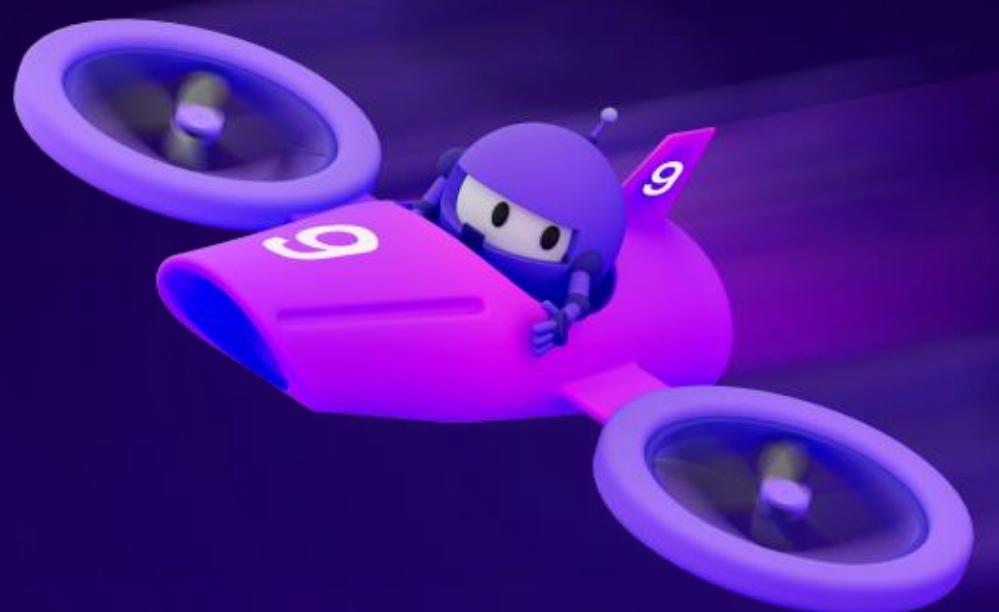


Beyond GitHub Copilot Tips and Tricks

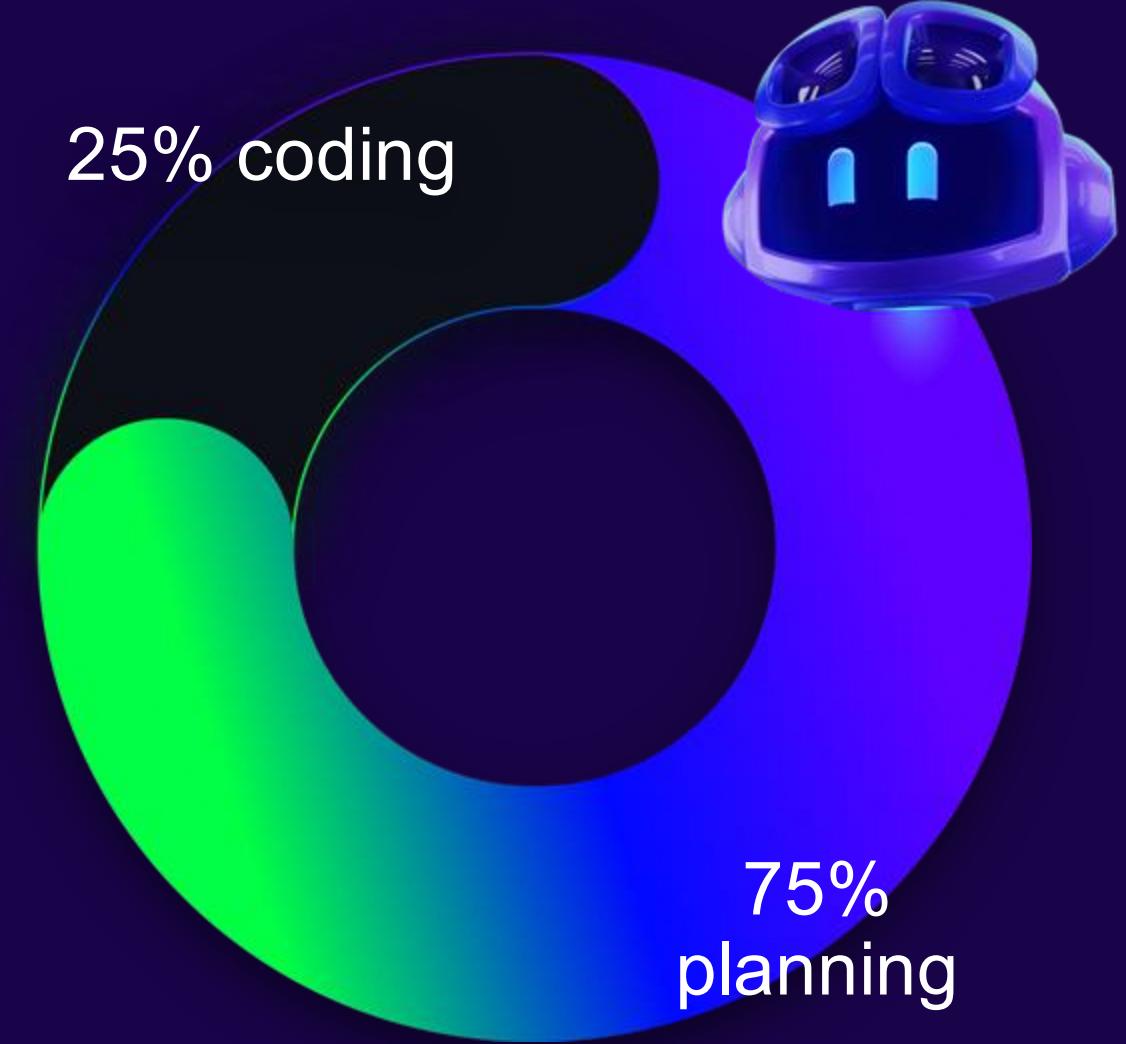
April Yoho

 Scubaninja

 /azureapril



Developers
dedicate 75%
to non-coding
tasks



GitHub Platform

The AI Powered Developer Platform
to Build, Scale, and Deliver Secure
Software





Hello!





We are building
Copilot for the sake of
developer happiness



Built for developer happiness



55%

faster

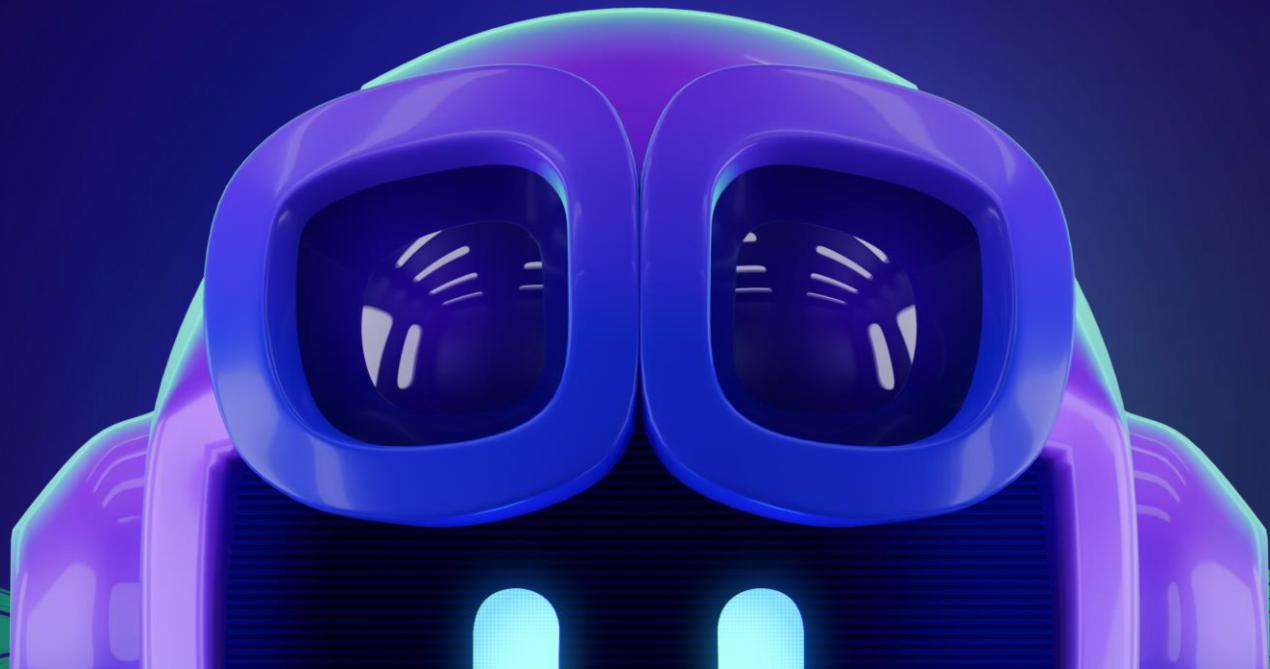
73%

were able to stay
in the flow

75%

felt more fulfilled
in their jobs

Coding in your natural language





Copilot Chat in VS & VS Code

GITHUB COPILOT: CHAT

GitHub Copilot

```
import unittest
import datetime

class TestParseExpenses(unittest.TestCase):
    def test_parse_expenses_with_valid_input(self):
        expenses_string = '2023-01-02 -34.01 USD\n2023-01-03 2.59 DKK
                           2023-01-03 -2.72 EUR'
        expected_output = [(datetime.datetime(2023, 1, 2),
                            -34.01, 'USD'),
                           (datetime.datetime(2023, 1, 3),
                            2.59, 'DKK'),
                           (datetime.datetime(2023, 1, 3),
                            -2.72, 'EUR')]
        self.assertEqual(parse_expenses(expenses_string),
                        expected_output)

    def test_parse_expenses_with_empty_input(self):
        expenses_string = ''
        expected_output = []
        self.assertEqual(parse_expenses(expenses_string),
                        expected_output)

    def test_parse_expenses_with_comments(self):
        expenses_string = '# This is a comment\n2023-01-02 -34.01 USD
                           2023-01-03 2.59 DKK
                           2023-01-03 -2.72 EUR'
        expected_output = [(datetime.datetime(2023, 1, 2),
                            -34.01, 'USD'),
                           (datetime.datetime(2023, 1, 3),
                            2.59, 'DKK'),
                           (datetime.datetime(2023, 1, 3),
                            -2.72, 'EUR')]
        self.assertEqual(parse_expenses(expenses_string),
                        expected_output)

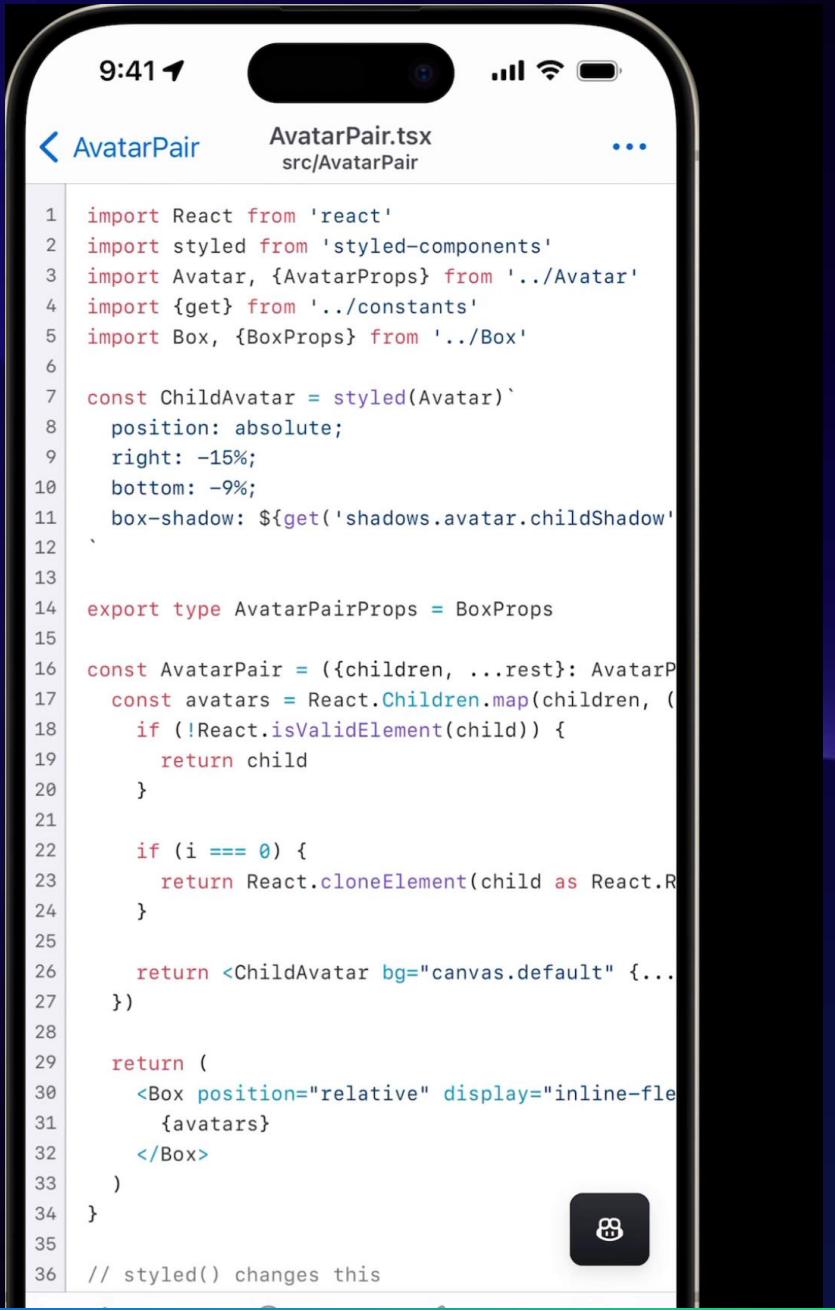
if __name__ == '__main__':
    unittest.main()
```

The code assumes that the `datetime` module is imported. The test cases cover different scenarios such as valid input, empty input, input with comments, invalid date format, and invalid value format.

Analyze and debug

Ask a question or type '/' for topics ➤

Copilot Chat in Mobile



A mobile phone screen showing a code editor interface. The top status bar indicates the time is 9:41 and shows signal, Wi-Fi, and battery icons. The title bar says "AvatarPair" with a back arrow, "AvatarPair.tsx", and a three-dot menu icon. The code editor displays the following TypeScript code:

```
1 import React from 'react'
2 import styled from 'styled-components'
3 import Avatar, {AvatarProps} from '../Avatar'
4 import {get} from '../constants'
5 import Box, {BoxProps} from '../Box'
6
7 const ChildAvatar = styled(Avatar)`
8   position: absolute;
9   right: -15%;
10  bottom: -9%;
11  box-shadow: ${get('shadows.avatar.childShadow')}
12 `
13
14 export type AvatarPairProps = BoxProps
15
16 const AvatarPair = ({children, ...rest}: AvatarProps) => {
17   const avatars = React.Children.map(children, (child, i) => {
18     if (!React.isValidElement(child)) {
19       return child
20     }
21
22     if (i === 0) {
23       return React.cloneElement(child as React.ReactElement, rest)
24     }
25
26     return <ChildAvatar bg="canvas.default" {...rest}>{child}</ChildAvatar>
27   })
28
29   return (
30     <Box position="relative" display="inline-flex" gap={10}>
31       {avatars}
32     </Box>
33   )
34 }
35
36 // styled() changes this
```



Demo

Recap



Limitations



Prompts and context



Trust but verify

Technical Preview

Copilot Workspace

The Copilot-native dev environment, designed for everyday tasks

GitHub Copilot Workspace

Spec



Plan



Code



Build



Preview



Deploy



Issue



Repos



Codespaces



Actions



PR



Actions



GitHub Advanced Security





Demo: GitHub Copilot Workspace

PUBLIC PREVIEW

Copilot Extensions



[Code](#)[main](#) [+](#) [🔍](#)[Jump to file](#)[models](#)[_init_.py](#)[book.py](#)[user.py](#)[templates](#)[static](#)[database](#)[test](#)[app.py](#)[config.py](#)[requirements.txt](#)[README.md](#)[bookstore_app / models / book.py](#)[monalisa initial setup and book crud](#) [bc07c5d · 23 days ago](#) [History](#)[Code](#)[Blame](#)[6339 lines \(6339 sloc\) · 1.05 MB](#)[Raw](#)

```
1  class Book:
2      def __init__(self, isbn, title, author):
3          self.isbn = isbn
4          self.title = title
5          self.author = author
6
7      def __repr__(self):
8          return f"<Book(isbn='{self.isbn}', title='{self.title}', author='{self.author}')>"
9
10     @classmethod
11     def find_books_by_author(cls, session, author_name):
12         """
13             List all books by a given author, ordered by title. Return a list of Book objects.
14         """
15         rows = session.execute('SELECT * FROM books WHERE author_name = %s ORDER BY title', [author_name])
16         books = []
17         for row in rows:
18             books.append(cls(row.isbn, row.title, row.author_name))
19         return books
20
21     @classmethod
22     def find_by_isbn(cls, session, isbn):
23         """
24             Retrieve a book by its ISBN. Return a Book object or None if not found.
25         """
26         row = session.execute('SELECT * FROM books WHERE isbn = %s', [isbn]).one()
27         if row:
28             return cls(row.isbn, row.title, row.author_name)
```

Recap



Build your internal community



Share tips, tricks and surprising moments



Iterate, iterate, iterate

Compliance & Security Questions

GitHub Copilot Trust Center

<https://gh.io/ctc>

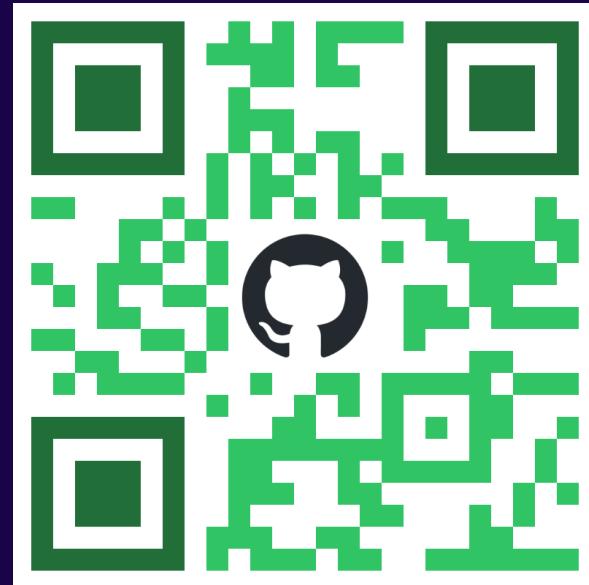
Resources

GitHub Copilot Trust Center



gh.io/TrustCenter

GitHub Interactive Courses



gh.io/skills

GitHub Learning Pathways



gh.io/learning-pathways

Thank you

