



# GitHub Copilot

(AI Pair Programming)



**GitHub**  
Copilot



# Contents

- AI Pair Programming
- GitHub
  - GitHub Pro
  - GitHub Enterprise
- GitHub Copilot
  - Copilot Individual
  - Copilot Enterprise



# AI Pair Programming

- GitHub Copilot
  - Gemini Code Assist
  - CodiumAI
  - Gemini in Android Studio (Studio Bot)
- 
- <https://www.codium.ai/codiumai-vs-github-copilot/>
  - <https://www.kajetandomagala.com/codeium-vs-copilot/>
  - <https://developer.android.com/studio/preview/gemini>



# GitHub (Basic)

- <https://medium.com/@naruapon/การใช้งาน-git-8a563531d1f1>



# GitHub Pro

- GitHub Support via email
- 3,000 GitHub Actions minutes per month
- 2 GB GitHub Packages storage
- 180 GitHub Codespaces core hours per month
- 20 GB GitHub Codespaces storage per month
- Advanced tools and insights in private repositories

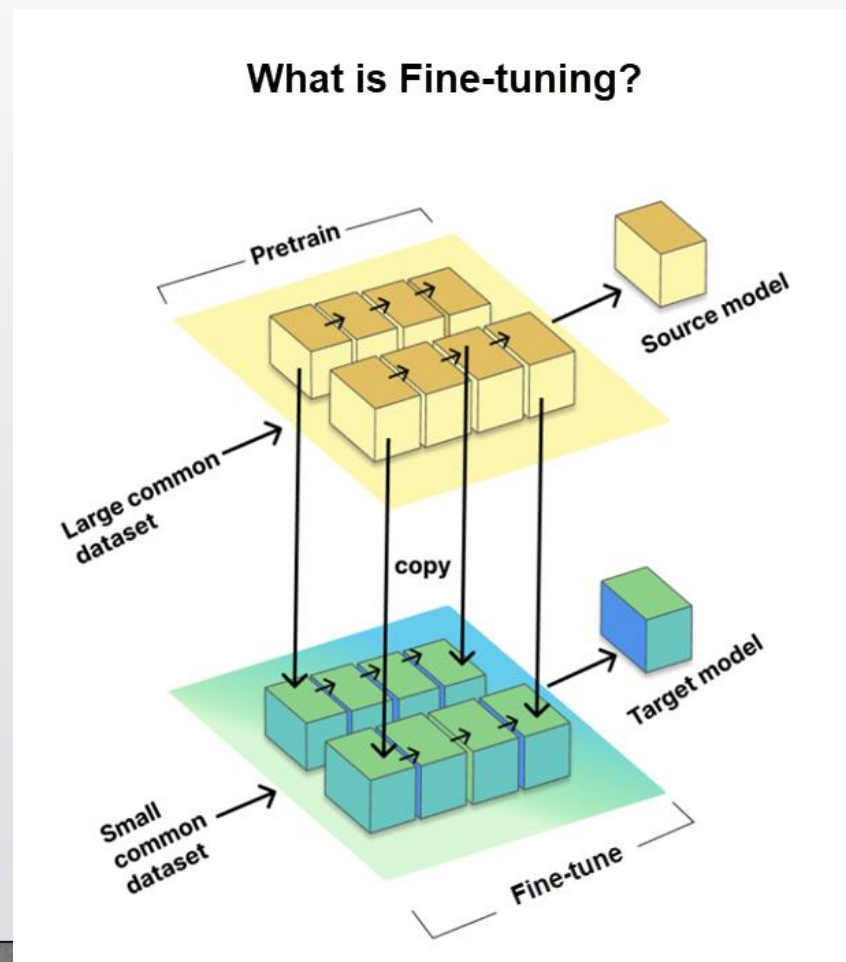


# How GitHub Copilot Works

- GitHub Copilot is powered by OpenAI Codex. The auto-generated suggestions come from the context within the file, like function names, code comments, docstrings, file names, cursor position, and more. Based on this information, Copilot suggests code snippets that developers can accept by simply pressing the Tab key on their keyboards.
- The AI tool understands TypeScript, Python, JavaScript, Ruby, and dozens of other common languages.



# GitHub Copilot Large Language Models (LLMs)









## Copilot Individual

- Who's it for: Individual developers, freelancers, students, and educators who want to code faster.
- Cost: \$10 per month or \$100 per year.
- Free for verified students, teachers, and maintainers of popular open-source projects.



## What's included:

- Chat
  - Unlimited messages and interactions
  - Context-aware coding support and explanations
  - Debugging and security remediation assistance
- Code completion
  - Real-time code suggestions
  - Comments to code



## What's included:

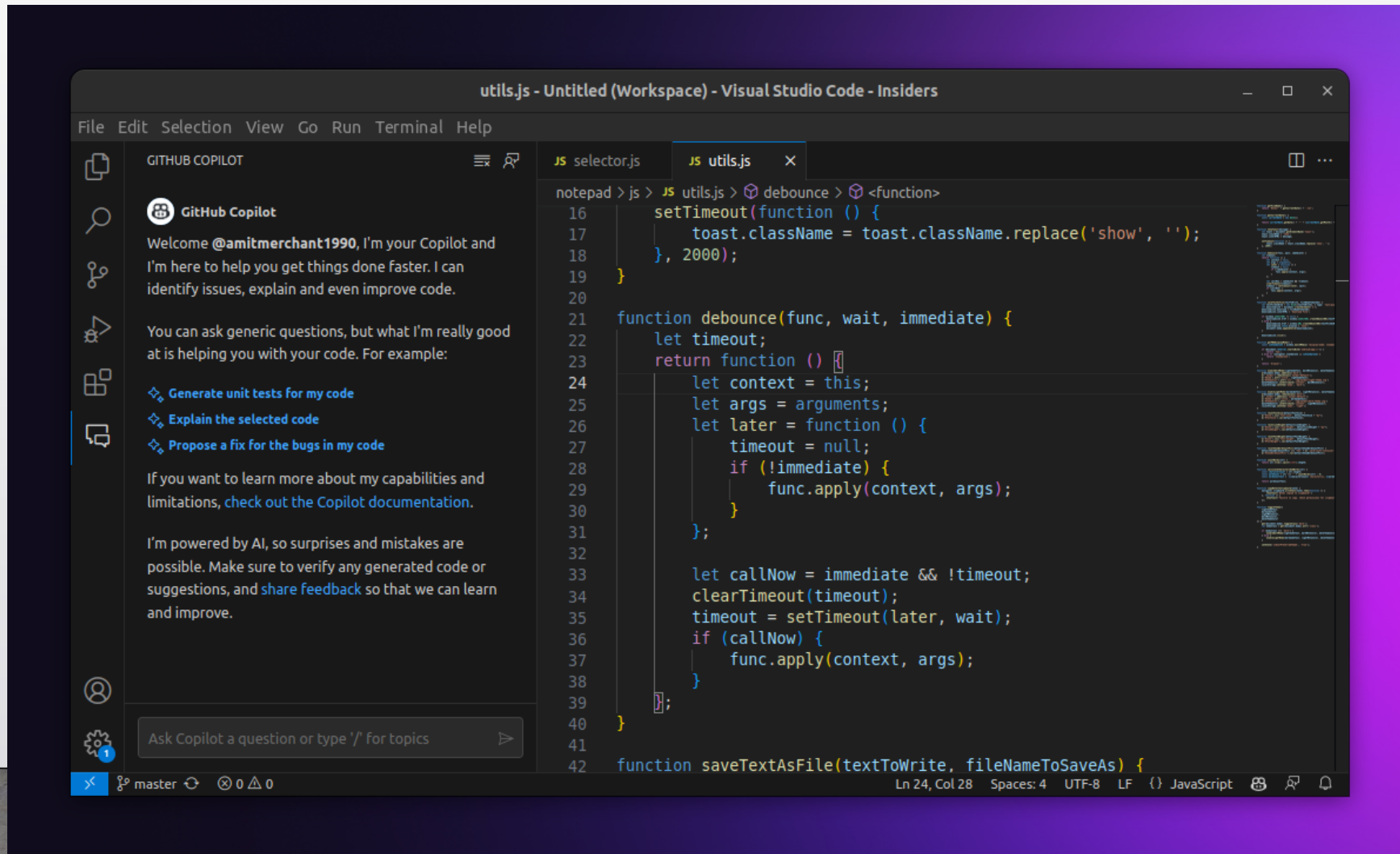
- Smart actions
  - Inline chat and prompt suggestions
  - Slash commands and context variables
  - Commit message generation
- Supported environments
  - IDE, CLI, and GitHub Mobile
- Management and policies
  - Public code filter



# Copilot Enterprise

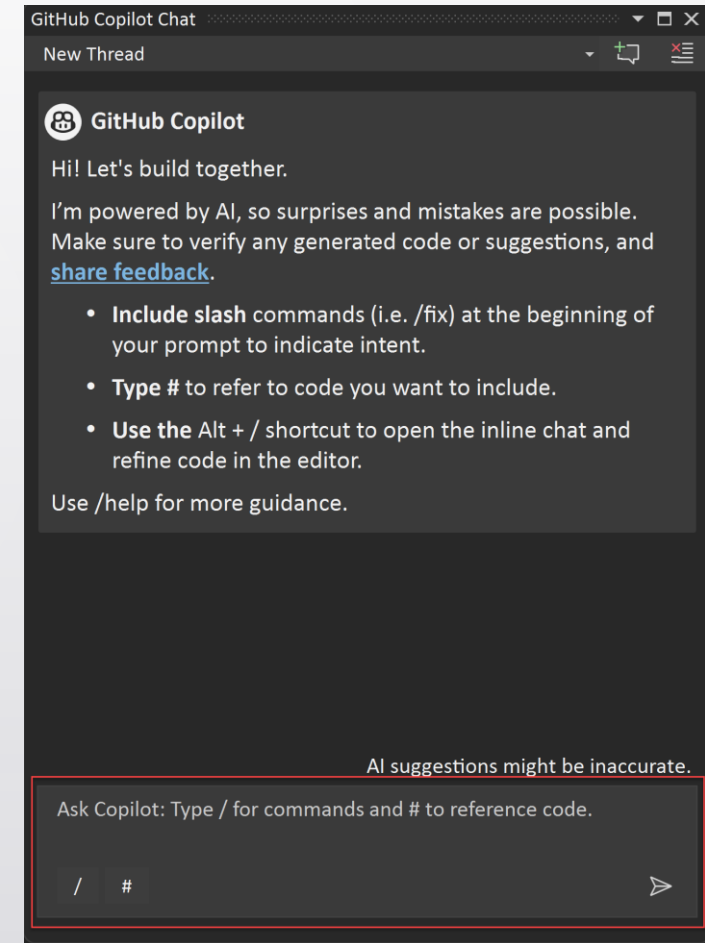
- **Who's it for:** Companies that want to customize GitHub Copilot to their organization and infuse AI across the developer workflow.
- **Cost:** \$39 per user per month.

# GitHub Copilot Chat (Visual Studio Code)



# Ask questions in the chat window

- In Visual Studio, select View > GitHub Copilot Chat.



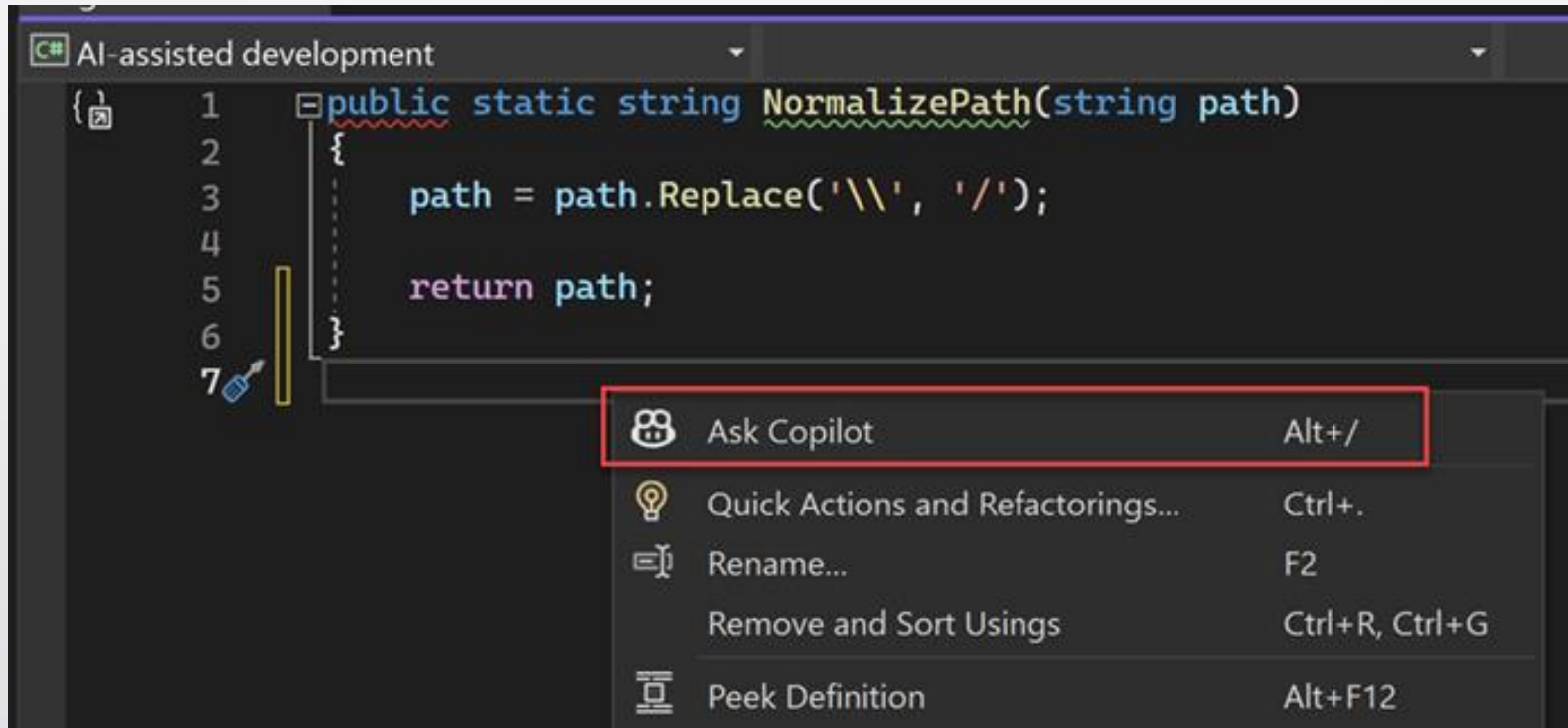




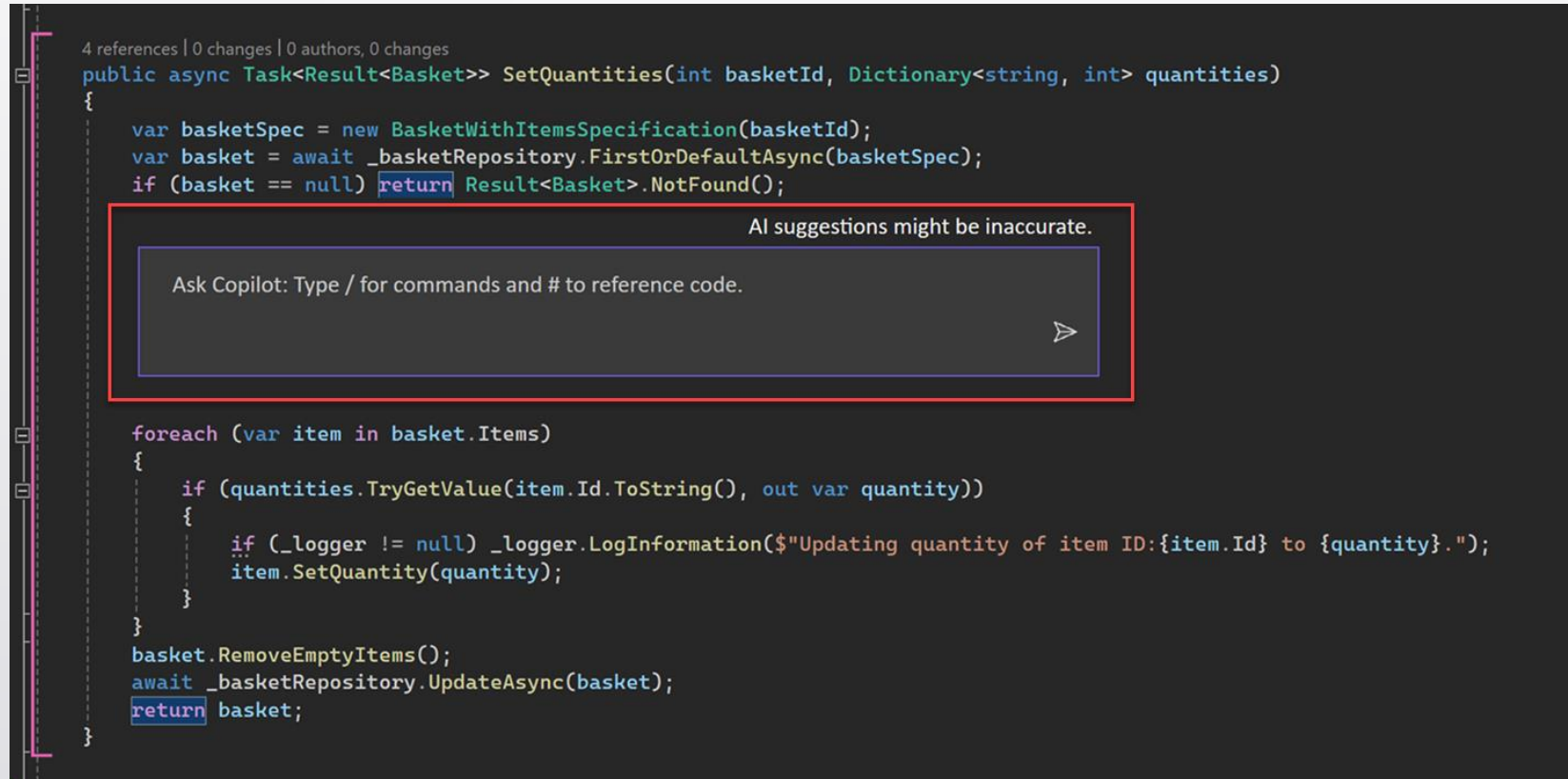
# GitHub Copilot in Visual Studio

- Inline Chat View
  - Ctrl+I (Visual Studio Code)
  - Alt + / (Visual Studio)

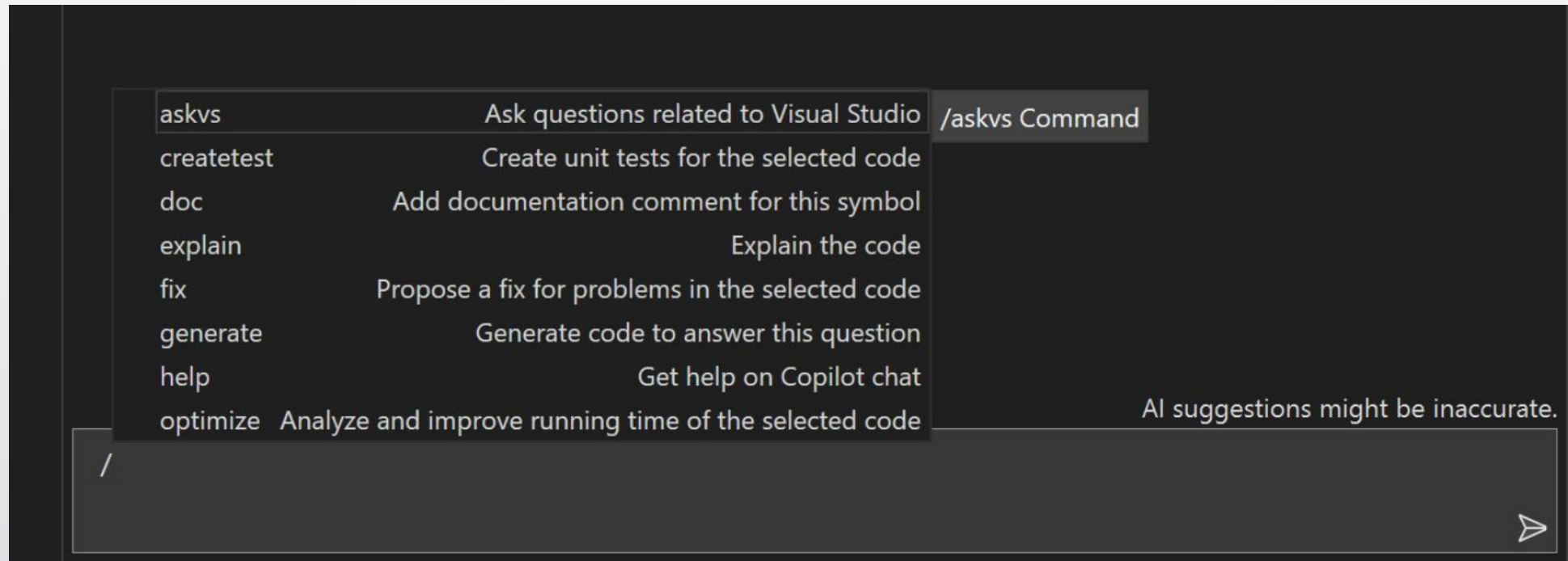
# Inline Chat View



# Inline Chat View



# Slash Commands to State Intent



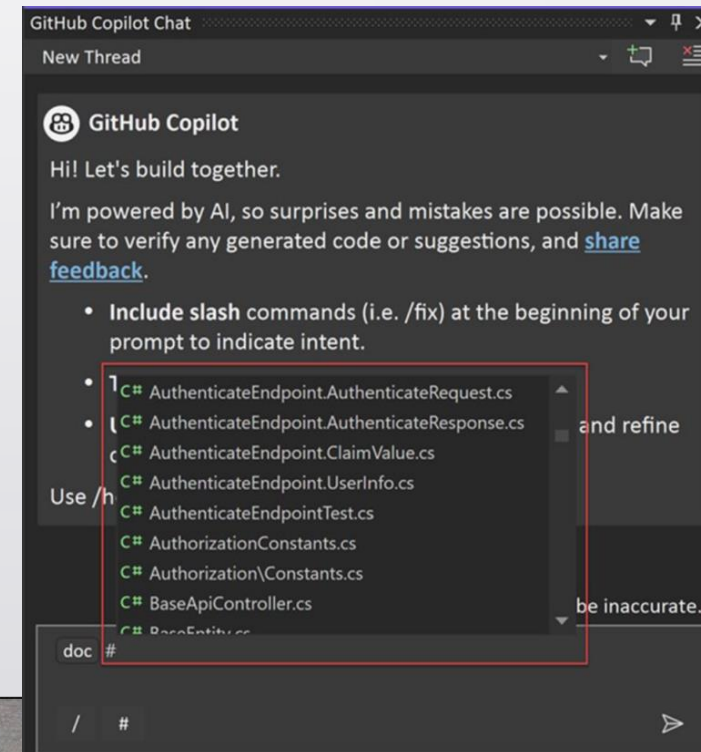
# Chat Window and Inline Chat

Command	Usage	Chat window	Inline chat
/doc	Add comments for specified or selected code. Examples: <ul style="list-style-type: none"><li>- /doc DeleteBasketAsync method in BasketService.cs</li><li>- select desired code and enter /doc</li></ul>	Yes	Yes
/explain	Get code explanations.  Examples: <ul style="list-style-type: none"><li>- /explain the AddItemToBasket method in BasketService.cs</li><li>- select desired code and enter /explain</li></ul>	Yes	Yes
/fix	Propose a fix for problems in the selected code. Examples: <ul style="list-style-type: none"><li>- /fix the SetQuantities method in BasketService.cs</li><li>- select desired code and enter /fix</li></ul>	Yes	Yes
/generate	Generate code to answer specified question. Example: /generate code to add two numbers in Calculator.cs	Yes	Yes



# Context Variables to Refine Your Scope

- The context variables feature allows you to specify files from your solution in your questions by using the # symbol.





# Analyze and Fix Test Window Failures

The screenshot displays the Visual Studio Test Explorer interface. At the top, a toolbar shows 5 failed tests (red X), 0 passed tests (green checkmark), and 5 tests in progress (yellow lightning bolt). Below the toolbar, a table lists the test results:

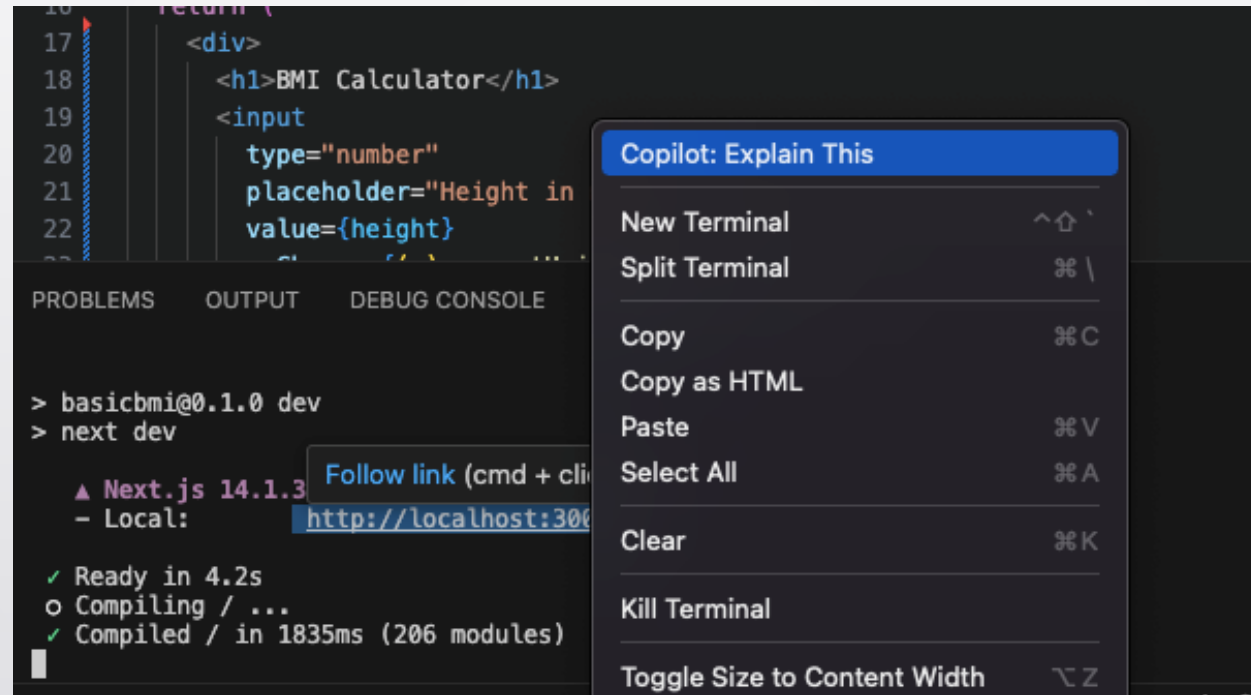
Test	Duration	Traits	Err...
MsTestSamples (5)	199 ms		
MsTestSamples (5)	199 ms		
UnitTest1 (5)	199 ms		
a_SimpleTest	158 ms		Ass...
b_IsEvenTest	< 1 ms		Ass...
c_GetNumberTest	1 ms		Test...
d_GetStringMiddleCharTest	40 ms		Test...
e_GoGetPartTest	< 1 ms		Test...

The right pane, titled "Test Detail Summary", provides information for the selected test, `c_GetNumberTest`:

- Source: `UnitTest1.cs` line 32
- Duration: 1 ms
- Message: Test method `MsTestSamples.UnitTest1.c_GetNumberTest` threw exception `System.DivideByZeroException: Attempted to divide by zero.`
- Stack Trace:
  - `Helper.GetANumber(Int32 v1, Int32 v2)` line 64
  - `Helper.GetANumber()` line 60
  - `UnitTest1.c_GetNumberTest()` line 35

An "Ask Copilot" button is visible at the bottom of the details pane.

# Explain terminal





# Github Copilot @workspace

- Github Copilot @workspace — New Development Experience
- <https://medium.com/@yar.dobroskok/github-copilot-workspace-new-development-experience-d69857fbd067>



yardobr

how do I start the current project?



GitHub Copilot

To start a project in Visual Studio Code, you typically need to follow these steps:

1. Open the terminal in Visual Studio Code. You can do this by going to the top menu and selecting `View -> Terminal`, or by using the shortcut `Ctrl+`` (backtick).
2. Navigate to the project's directory. You can do this by typing `cd path/to/your/project` in the terminal, replacing `path/to/your/project` with the actual path to your project.
3. Depending on the type of your project, you will need to run a specific command to start it. Here are some common examples:
  - For a Node.js project, you might run `npm start` or `node app.js` (replace `app.js` with the entry point to your application).
  - For a Python project, you might run `python main.py` (replace `main.py` with the main Python file in your project).
  - For a Java project, you might run `javac Main.java` followed by `java Main` (replace `Main` with the main class in your project).

Please replace the commands with the ones specific to your project. If you're not sure what command to run, check the documentation of the technology you're using or the README file of your project.



 yardobr

@workspace how do I start the current project?

 GitHub Copilot used @workspace



Used 8 references >

Used references to the actual files

Based on your workspace structure, it seems like your project is divided into three main parts: `classifier_server`, `client`, and `server`. Each of these parts likely needs to be started separately. Here's how you can do it:

1. **classifier\_server**: This appears to be a Python application. You can start it by navigating to the `classifier_server` directory and running the `classifier_api.py` script with Python. Make sure to install any necessary dependencies listed in the `requirements.txt` file first.

It now knows my actual project structure and understands the main building blocks

```
cd classifier_server
pip install -r requirements.txt
python classifier_api.py
```

2. **client**: This appears to be a JavaScript (possibly React) application. You can start it by navigating to the `client` directory, installing the necessary dependencies with `npm install`, and then starting the application with `npm start`.

```
cd client
npm install
npm start
```

3. **server**: This appears to be a Node.js application. You can start it by navigating to the `server` directory, installing the necessary dependencies with `npm install`, and then starting the application with `npm start`.

```
cd server
npm install
npm start
```

❖ How can I configure the admin account and create initial communities for the SocialEcho application?



# The Ultimate Manual to GitHub Copilot

- Example #1: Convert Comments to Code
- Example #2: Autofill Repetitive Code
- Example #3: Run Tests
- Example #4: Navigating Unfamiliar Territory
- Example #5: Creating an Application Entirely With Copilot



## Example #1: Convert Comments to Code

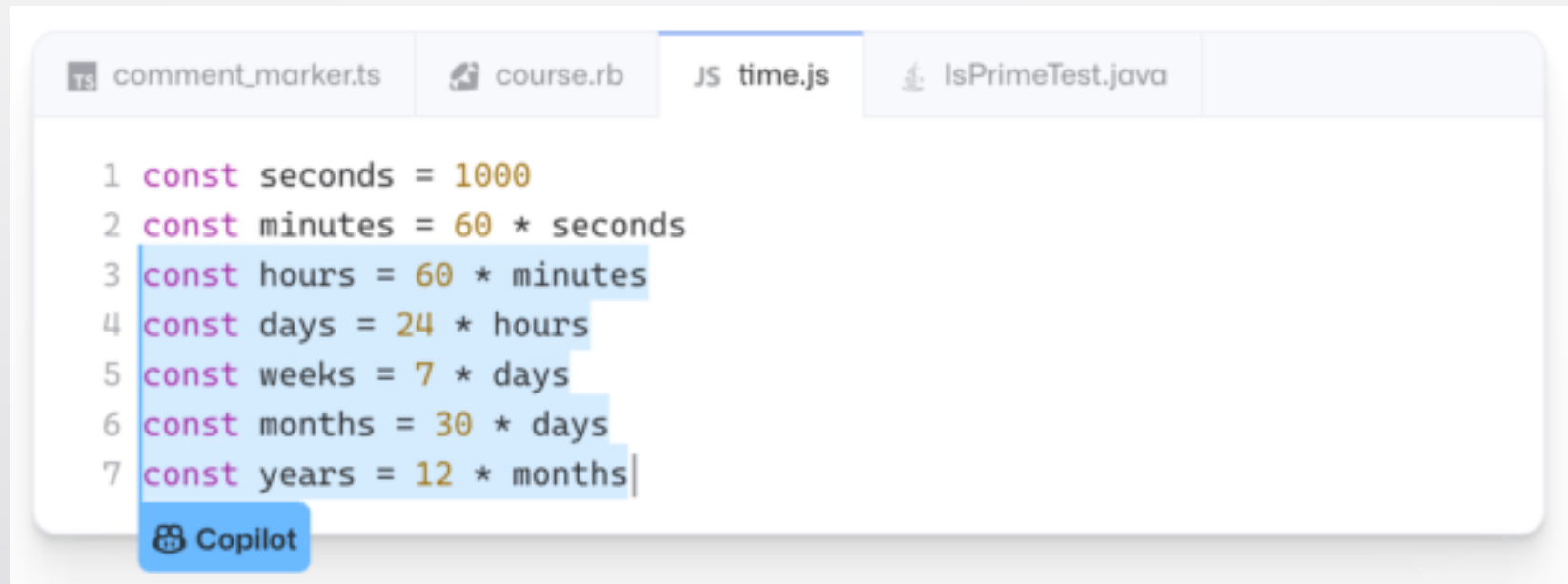


The image shows a code editor interface with four tabs: 'collaborators.ts', 'get\_repositories.py', 'JS non\_alt\_images.js', and 'PersonUtils.java'. The 'get\_repositories.py' tab is active, displaying a Python script. The script is as follows:

```
1 import urllib.request, json
2
3 def get_repositories(org):
4     """List all names of GitHub repositories for an org."""
5     url = 'https://api.github.com/orgs/' + org + '/repos'
6     request = urllib.request.Request(url)
7     response = urllib.request.urlopen(request)
8     data = json.loads(response.read().decode())
9     return [repo['name'] for repo in data]
```

A blue Copilot icon is visible in the bottom left corner of the code editor window.

## Example #2: Autofill Repetitive Code



The screenshot shows a code editor with four tabs: 'comment\_marker.ts', 'course.rb', 'JS time.js', and 'IsPrimeTest.java'. The 'JS time.js' tab is active. The code in the editor is as follows:

```
1 const seconds = 1000
2 const minutes = 60 * seconds
3 const hours = 60 * minutes
4 const days = 24 * hours
5 const weeks = 7 * days
6 const months = 30 * days
7 const years = 12 * months|
```

A blue highlight covers the repetitive pattern of the code from line 3 to line 7. A blue button with the Copilot logo and the text 'Copilot' is positioned at the bottom left of the code editor area.

## Example #3: Run Tests

```
strip_suffix.py  gcd.rb  JS count_button.js

1 import React from "react";
2 import { render, fireEvent } from "@testing-library/react";
3
4 function Counter() {
5   const [count, setCount] = React.useState(0);
6   return (
7     <div>
8       <button onClick={() => setCount(currCount => currCount + 1)}>
9         Increment
10       </button>
11       <p>Count: {count}</p>
12     </div>
13   );
14 }
15
16 // a unit test that asserts that count increases when the button is clicked
17 it("increments count", () => {
18   const { getByText } = render(<Counter />);
19   const button = getByText("Increment");
20   fireEvent.click(button);
21   expect(getByText("Count: 1")).toBeInTheDocument();
22 });
```

Copilot

## Example #4: Navigating Unfamiliar Territory

```
JS draw_scatterplot.js  draw_scatterplot.py  draw_scatterplot.rb  draw_scatterplot.ts

1 import matplotlib.pyplot as plt
2
3 def draw_scatterplot(x_values, y_values):
4     plt.scatter(x_values, y_values, s=20)
5     plt.title("Scatter Plot")
6     plt.xlabel("x values")
7     plt.ylabel("y values")
8     plt.show()
```

Copilot



# How to Get Started With GitHub Copilot

- Step 1: Narrow Down Your Use Case and Goal
- Step 2: Install the GitHub Copilot Extension
- Step 3: Learn the GitHub Copilot Keyboard Shortcuts
- Step 4: Start Writing Your Code and Review the Suggestions
- Step 5: Make Edits and Test Your Code



## Step 2: Install the GitHub Copilot Extension

- GitHub Copilot — Visual Studio Code Marketplace
- GitHub Copilot — JetBrains Marketplace
- Neovim Plugin for GitHub Copilot





# GitHub Copilot — Visual Studio Code Marketplace

- <https://marketplace.visualstudio.com/items?itemName=GitHub.copilot>



# How to Install GitHub Copilot in Visual Studio

- A GitHub account. You can sign up for free at [GitHub.com](https://github.com).
- An active GitHub Copilot subscription (Individual, Business, or Enterprise).
- Visual Studio 2022 17.6 or later installed.



# How to Install GitHub Copilot in Visual Studio Code

- In the Visual Studio Code Marketplace, go to the GitHub Copilot extension page and click Install.
- A popup will appear, asking to open Visual Studio Code. ...
- In the "Extension: GitHub Copilot" tab in Visual Studio Code, click Install.

# Install GitHub Copilot in Visual Studio Code

The screenshot displays the Visual Studio Code interface with the Extensions Marketplace open. The search bar at the top left contains the text "github copilot". The left sidebar lists several extensions, with "GitHub Copilot" (v1.180.0) highlighted in blue. The main panel shows the details for the "GitHub Copilot" extension, including its icon, name, version, and a description: "Your AI pair programmer". The extension has 15,403,202 downloads and a 4.5-star rating from 1020 reviews. The "Install" button is visible. Below the extension details, there are tabs for "DETAILS", "FEATURES", "CHANGELOG", and "EXTENSION PACK". The "DETAILS" tab is active, showing a section titled "Your AI pair programmer" and a code snippet. The code snippet is a Python function that finds the common prefix of two lists. The right sidebar contains sections for "Categories" (Programming Languages, Snippets, Machine Learning, Education), "Resources" (Marketplace, Issues, GitHub), and "More Info" (Published: 2021-06-29, 21:26:17; Last released: 2024-04-18, 08:27:52).

EXTENSIONS: MARKETPLACE

github copilot

GitHub Copilot v1.180.0

GitHub Integrates github and its w... KnisterPeter

GitHub Co... 15.4M ★ 3.5 Your AI pair programmer GitHub Install

GitHub Th... 12.4M ★ 4.5 GitHub theme for VS Code GitHub Install

GitHub Pull ... 20.3M ★ 3 Pull Request and Issue Pro... GitHub Install

GitHub Code... 3.9M ★ 5 Your instant dev environm... GitHub Install

GitHub Copil... 9.6M ★ 3 AI chat features powered ... GitHub Install

GitHub Rep... 2.6M ★ 4.5 Remotely browse and edit ... GitHub Install

GitHub Acti... 1.8M ★ 3.5 GitHub Actions workflows ... GitHub Install

Extension: GitHub Copilot

GitHub Copilot v1.180.0

GitHub github.com 15,403,202 ★★★★★ (1020)

Your AI pair programmer

Install

DETAILS FEATURES CHANGELOG EXTENSION PACK

Followers 37k Follow @github Views 16M Try Copilot Free trial

Your AI pair programmer

Get Code Suggestions in real-time, right in your IDE

```
1 def common_prefix(a, b):
2     """Return the common prefix of two lists."""
3     if len(a) < len(b):
4         return common_prefix(b, a)
5     for i in range(len(a)):
6         if a[i] != b[i]:
7             return a[:i]
8     return a
9
10
11 def test_common_prefix():
12     assert common_prefix(['a', 'b', 'c'], ['a', 'b', 'c', 'd']) == ['a', 'b', 'c']
13     assert common_prefix(['a', 'b', 'c'], ['b', 'c', 'd']) == ['b', 'c']
14     assert common_prefix(['a', 'b', 'c'], ['b', 'c']) == ['b', 'c']
15     assert common_prefix(['a', 'b', 'c'], ['a', 'b']) == ['a', 'b']
16     assert common_prefix([], []) == []
17
```

Categories

Programming Languages

Snippets

Machine Learning

Education

Resources

Marketplace

Issues

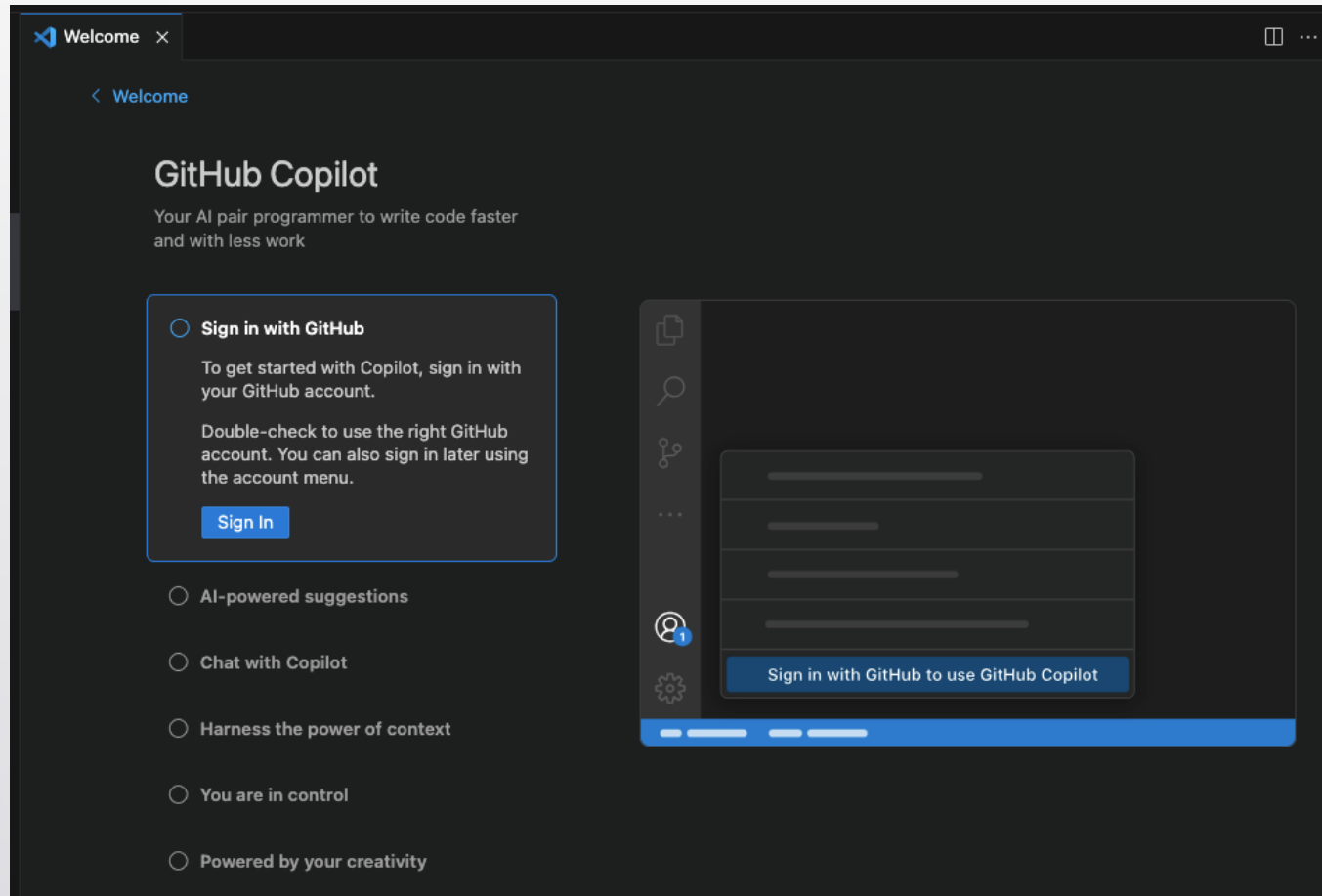
GitHub

More Info

Published 2021-06-29, 21:26:17

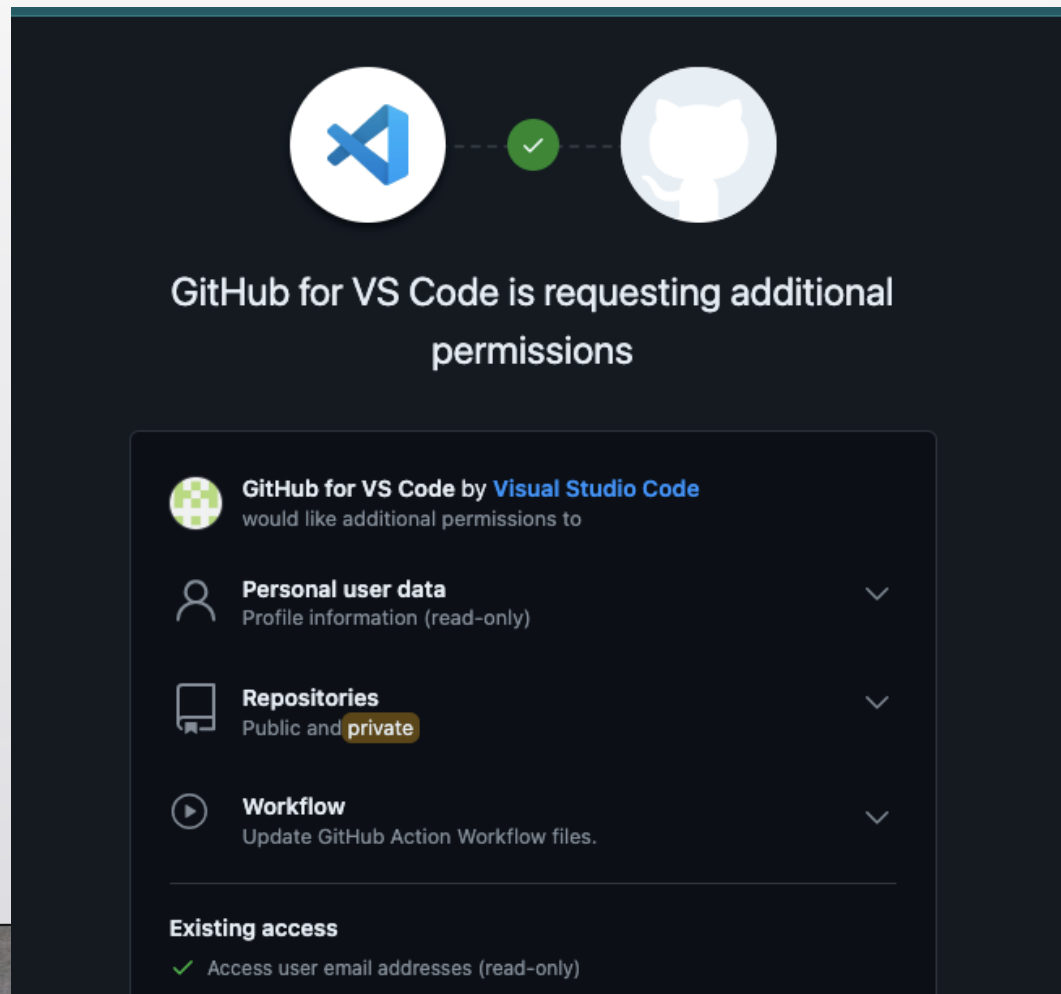
Last released 2024-04-18, 08:27:52

# Install GitHub Copilot in Visual Studio Code

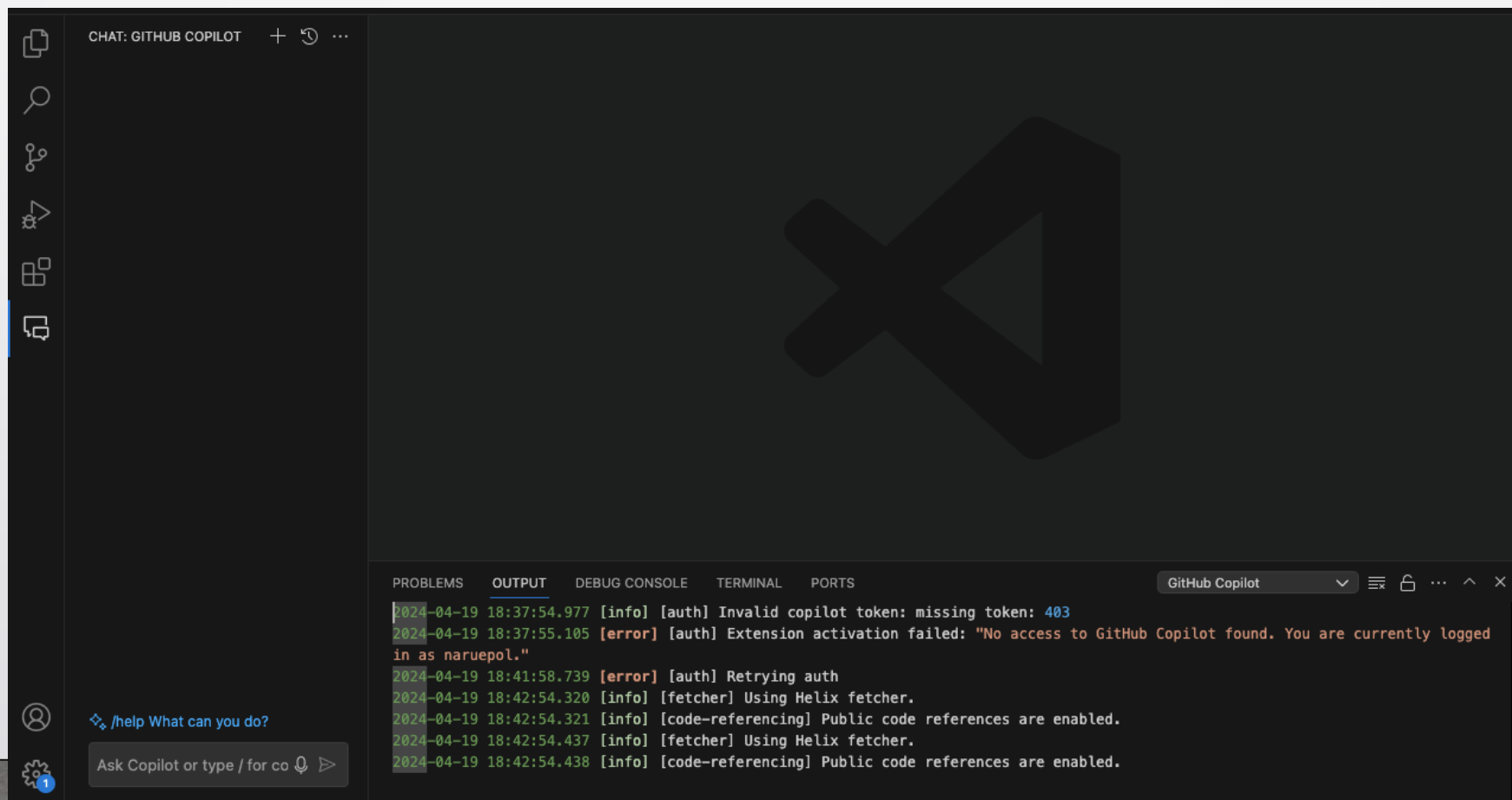




# Install GitHub Copilot in Visual Studio Code



# Chat GitHub Copilot



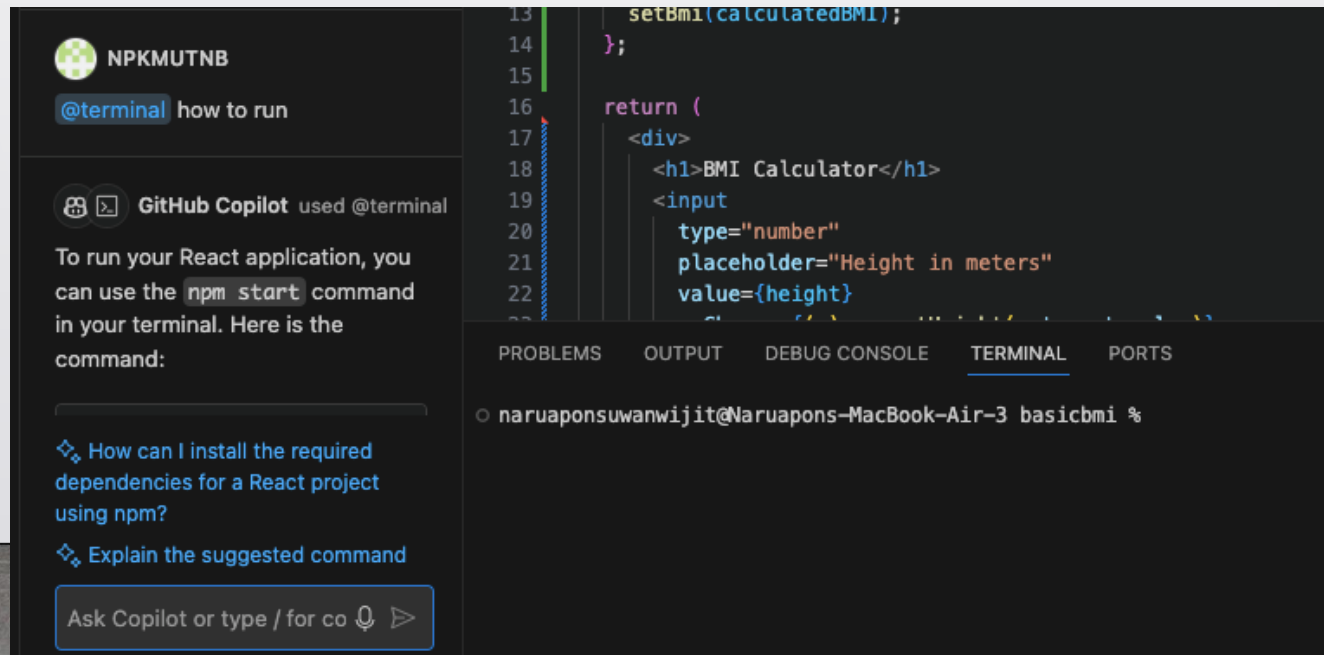


## Step 3: Learn the GitHub Copilot Keyboard Shortcuts

- Accept inline code suggestion — **Tab**
- Dismiss inline code suggestion — **Esc**
- Show next suggestion — **Alt + ]** or **Option (⌘) + ]**
- Show previous suggestion — **Alt + [** or **Option (⌘) + [**
- Trigger suggestion — **Alt + \** or **Option (⌘) + \**
- Open 10 suggestions in a separate pane — **Ctrl + Enter**

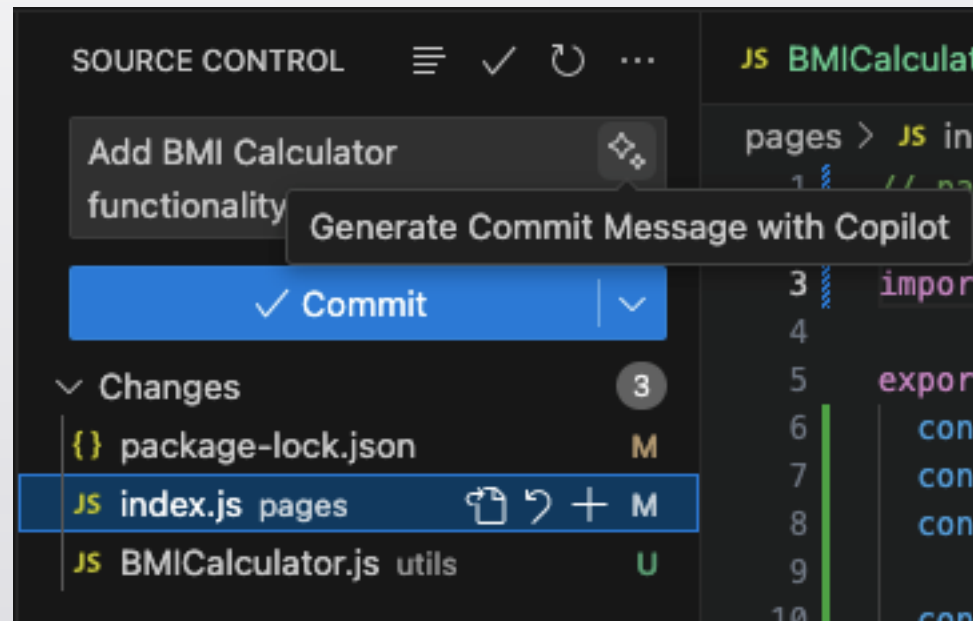
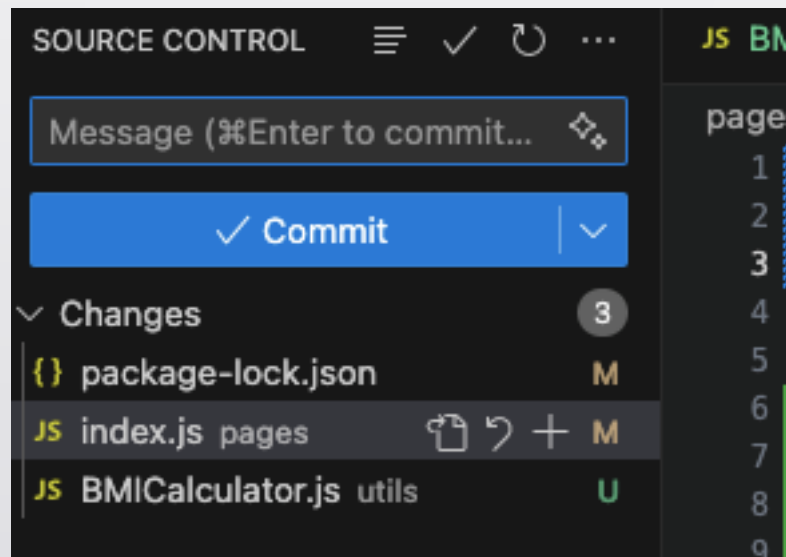
# Run terminal commands from GitHub Copilot Chat

- If you ever forget how to run a particular command when you're working in your VS Code, GitHub Copilot Chat is here to help! With the new `@terminal` agent in VS Code, you can ask GitHub Copilot how to run a particular command.



# Generate commit messages

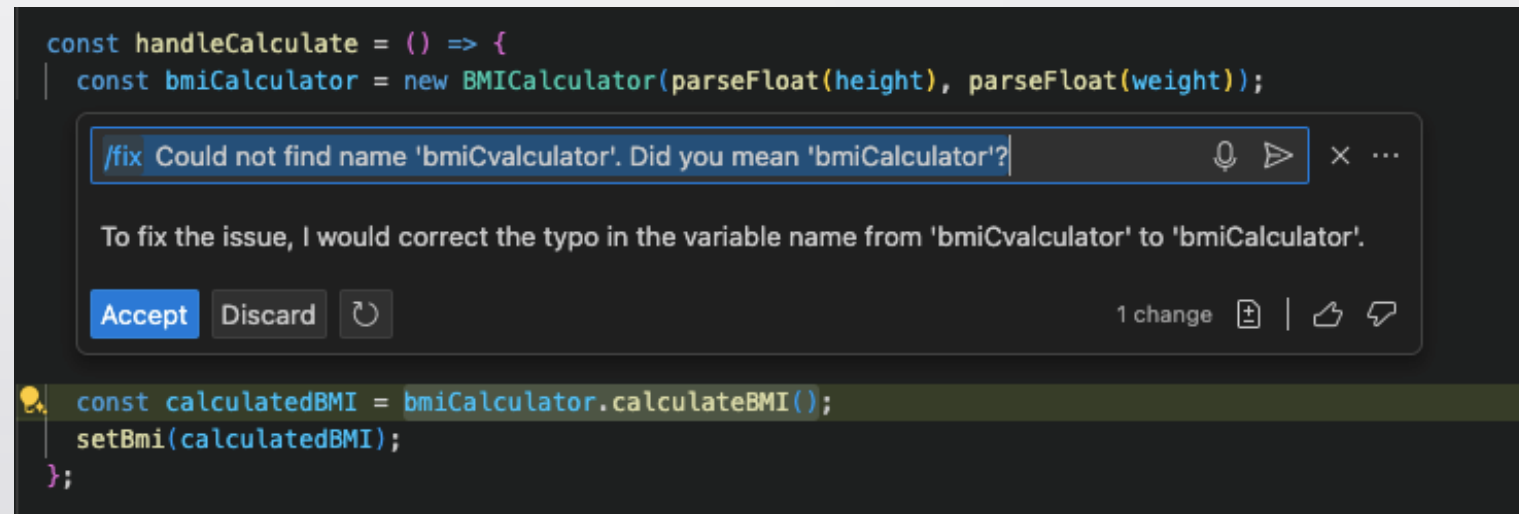
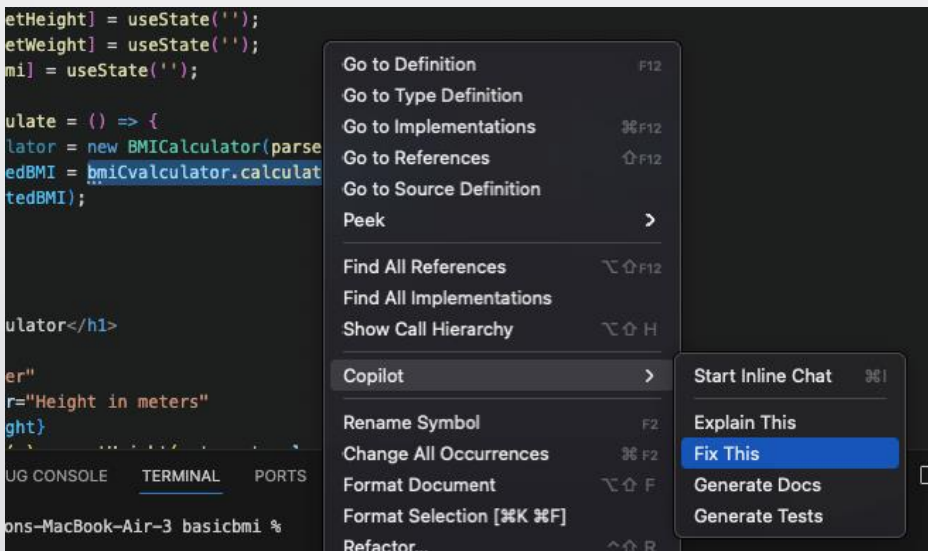
- GitHub Copilot can help you generate commit messages right in your IDE.





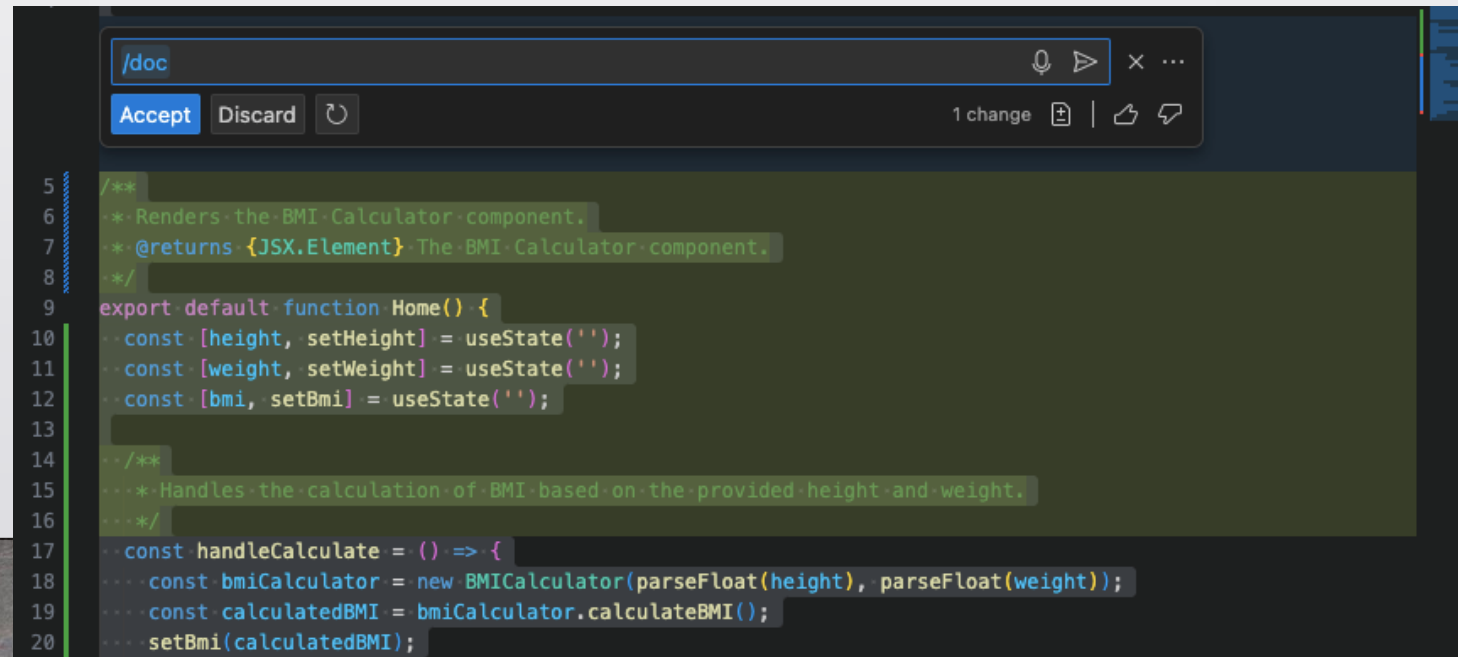
# Fix code inline

- highlight the code you want to fix, right click, and select “Fix using Copilot.”



# Generate documentation for your code

- `/doc`
- quickly generate documentation following language specific formats—Docstring for Python, JSDoc for Javascript or Javadoc for Java.

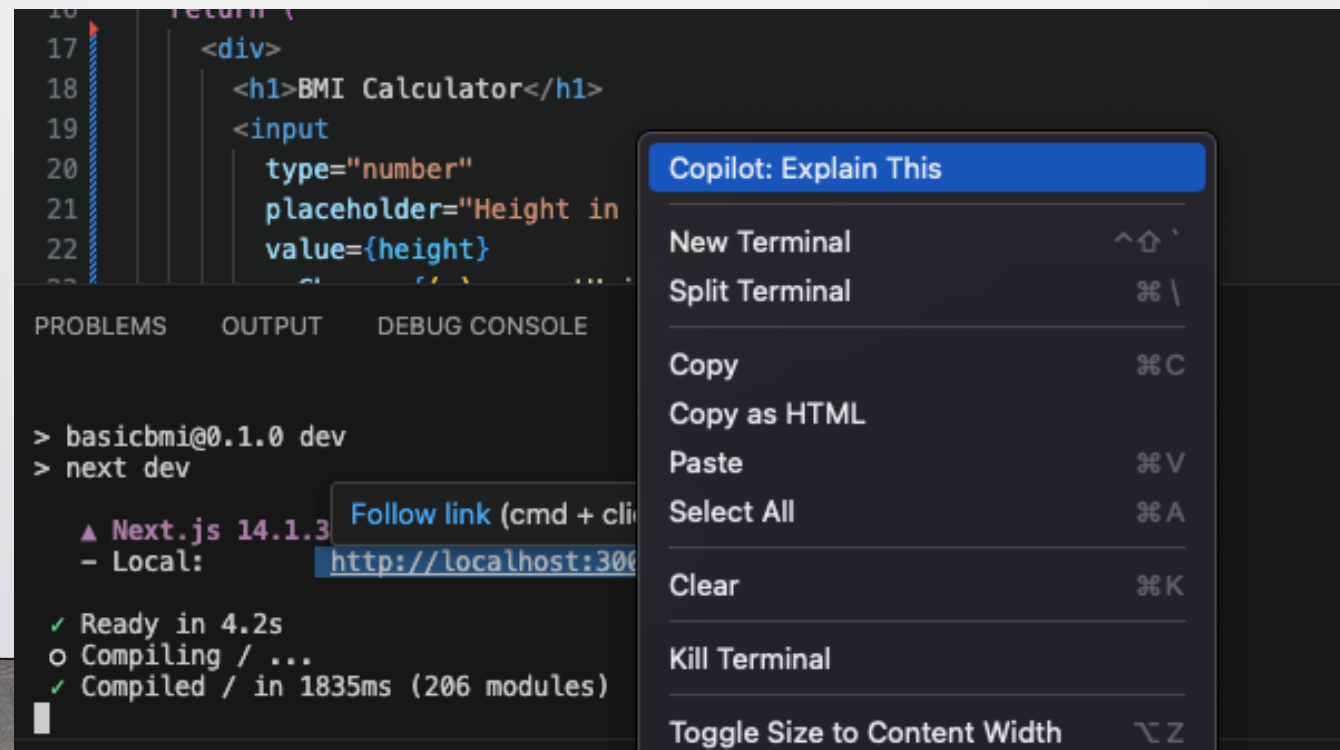


The screenshot shows a code editor interface. At the top, a command palette is open with the text `/doc` entered. Below the input field are buttons for 'Accept', 'Discard', and a refresh icon. To the right of these buttons, it says '1 change' followed by icons for undo, redo, and a link. The code editor below shows a JavaScript code snippet for a BMI calculator component. The code includes JSDoc comments and a function definition.

```
5  /**
6   * Renders the BMI Calculator component.
7   * @returns {JSX.Element} The BMI Calculator component.
8   */
9  export default function Home() {
10   const [height, setHeight] = useState('');
11   const [weight, setWeight] = useState('');
12   const [bmi, setBmi] = useState('');
13
14   /**
15    * Handles the calculation of BMI based on the provided height and weight.
16    */
17   const handleCalculate = () => {
18     const bmiCalculator = new BMICalculator(parseFloat(height), parseFloat(weight));
19     const calculatedBMI = bmiCalculator.calculateBMI();
20     setBmi(calculatedBMI);
```

# Get help with error messages in your terminal

- highlight the error message, right click, and select “Explain with Copilot.”





# Prompt engineering foundations and best practices

- What is prompt engineering?
- Foundations of prompt engineering
- Best practices in prompt engineering
- How Copilot learns from your prompts



# What is prompt engineering?

- Prompt engineering is the process of crafting clear instructions to guide AI systems, like GitHub Copilot, to generate context-appropriate code tailored to your project's specific needs.
- This ensures the code is syntactically, functionally, and contextually correct. Think of it like giving precise directions to a driver.
- Without them, the journey might be inefficient. But with clear guidance, the route becomes direct and efficient, saving time and energy.





# Principles of prompt engineering (4 S's)

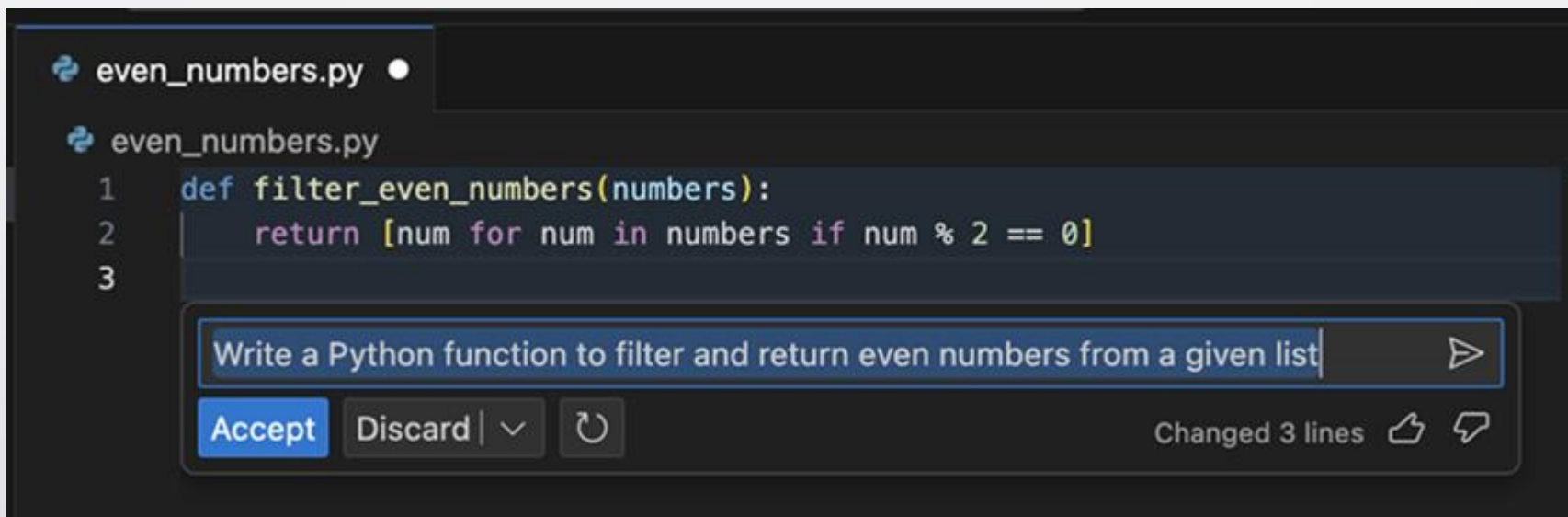
- Single: Always focus your prompt on a single, well-defined task or question. This clarity is crucial for eliciting accurate and useful responses from Copilot.
- Specific: Ensure that your instructions are explicit and detailed. Specificity leads to more applicable and precise code suggestions.
- Short: While being specific, keep prompts concise and to the point. This balance ensures clarity without overloading Copilot or complicating the interaction.
- Surround: Utilize descriptive filenames and keep related files open. This provides Copilot with rich context, leading to more tailored code suggestions.



# Best practices in prompt engineering

- Provide enough clarity
- Provide enough context with details
- Provide examples for learning

# Provide enough clarity



The screenshot shows a code editor with a file named `even_numbers.py`. The code defines a function `filter_even_numbers` that takes a list of numbers and returns a list of even numbers. Below the code, there is a chat interface with a text input field containing the prompt "Write a Python function to filter and return even numbers from a given list". The chat interface also includes buttons for "Accept", "Discard", and a refresh icon, along with a status indicator "Changed 3 lines".

```
even_numbers.py
1 def filter_even_numbers(numbers):
2     return [num for num in numbers if num % 2 == 0]
3
```

Write a Python function to filter and return even numbers from a given list

Accept Discard ↕ ↻ Changed 3 lines

# Provide enough context with details

```
even_numbers.py •
even_numbers.py
1  # write a simple flask app that returns a list of even numbers from a list of numbers
2  # Create a function that takes a list of numbers and returns only the even values.
3  # create a sample list of numbers
4  # create a list of even numbers from the sample list
5  # return the list of even numbers
6
7
```

# Provide examples for learning

```
even_numbers.py •
even_numbers.py
1  # create a sample list of numbers
2  # create a list of even numbers from the sample list
3  # return the list of even numbers
4  # Example: [1, 2, 3, 4, 5, 6, 7, 8, 9] -> [2, 4, 6, 8]
5
6  sample_list = [1, 2, 3, 4, 5, 6, 7, 8, 9]
```





# How Copilot learns from your prompts

GitHub Copilot operates based on AI models trained on vast amounts of data. To enhance its understanding of specific code contexts, engineers often provide it with examples. This practice, commonly found in machine learning, led to different training approaches such as:

- Zero-shot learning
- One-shot learning
- Few-shot learning

# Zero-shot learning

- GitHub Copilot generates code without any specific examples, relying solely on its foundational training

```
1  # I need a function to convert Celsius to Fahrenheit
2
3  def celsius_to_fahrenheit(celsius):
4
5      fahrenheit = (celsius * 9/5) + 32
6      return fahrenheit
7
```

# One-shot learning

- Building upon the previous zero-shot example

```
# Example: Function to convert Fahrenheit to Celsius
def fahrenheit_to_celsius(fahrenheit):
    celsius = (fahrenheit - 32) * 5/9
    return celsius

# Now I need a function to convert Celsius to Fahrenheit

def celsius_to_fahrenheit(celsius):
    fahrenheit = celsius * 9/5 + 32
    return fahrenheit
```



# Few-shot learning

- In this method, Copilot is presented with several examples, which strike a balance between zero-shot unpredictability and the precision of fine-tuning.

# Few-shot learning

```
greeting.py x
greeting.py
1  # Example 1: Greeting message for morning
2  # Input: 9 AM
3  # Output: "Good morning!"
4
5  # Example 2: Greeting message for afternoon
6  # Input: 2 PM
7  # Output: "Good afternoon!"
8
9  # Example 3: Greeting message for evening
10 # Input: 7 PM
11 # Output: "Good evening!"
12
13 # Now, generate a python code that takes the current time as input using the datetime module
14 # and returns the appropriate greeting message
15
16 # Solution:
17 # Import datetime module
18 import datetime
19
20 # Get current time
21 current_time = datetime.datetime.now()
22
23 # Get current hour
24 current_hour = current_time.hour
25
26 # Check if it is morning (before 12 PM)
27 if current_hour < 12:
28     print("Good morning!")
29
30 # Check if it is afternoon (between 12 PM and 4 PM)
31 elif current_hour < 16:
32     print("Good afternoon!")
33
34 # Check if it is evening (after 4 PM)
35 elif current_hour < 21:
36     print("Good evening!")
37
38 # Else it is night time
39 else:
40     print("Good night!")
41
```





# Validating the output of GitHub Copilot

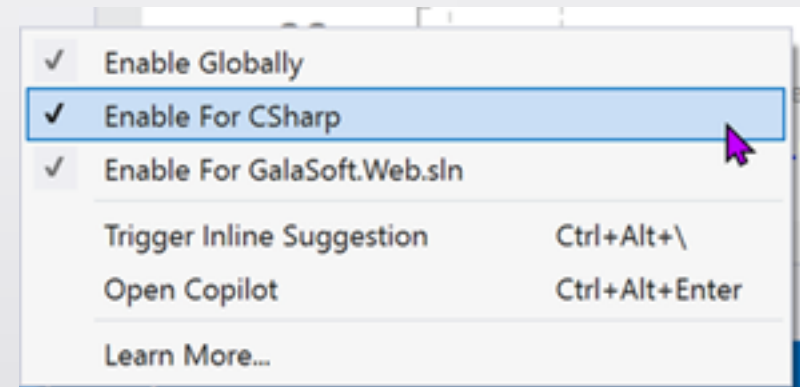
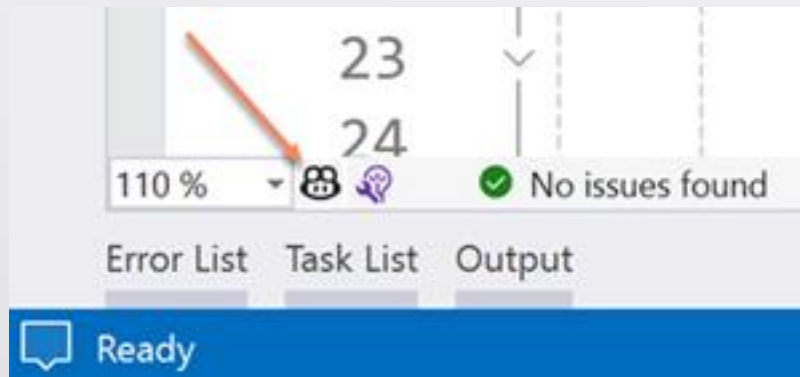
- The key principle to remember when working with AI assistants is that you should always verify the output. You have the chance to review the Copilot output either before accepting, when the code appears greyed out inline or after you have accepted the suggestion with the Tab key.

GitHub Copilot is not a compiler!

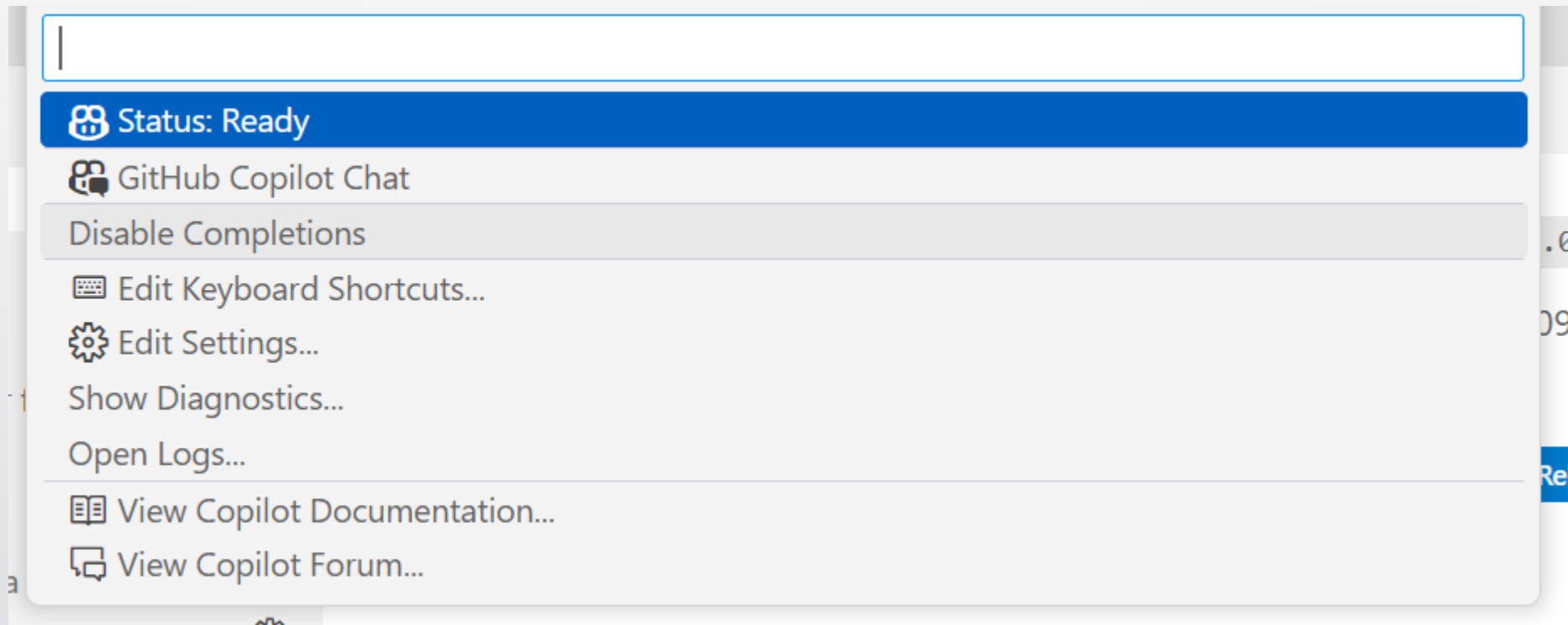
- This means that the code produced by GitHub Copilot can indeed fail to compile and/or be incorrect. It is your responsibility to review, resolve, and improve the suggested code, as necessary.



# Disabling GitHub Copilot completions (VS Studio)




# Disabling GitHub Copilot completions (VS Code)



# Disabling GitHub Copilot

- ctrl-shift-x : right click copilot - disable

Extension: GitHub Copilot X



## GitHub Copilot

v1.180.0

GitHub [github.com](#) | 15,438,095 | ★★★★★ (1020)

Your AI pair programmer

[Disable](#) [Uninstall](#) [Switch to Pre-Release Version](#) ⚙️

This extension is enabled globally.

DETAILS FEATURES CHANGELOG EXTENSION PACK

Followers 38k Follow @github Views 16M

Try Copilot Free trial

Your AI pair programmer

Get Code Suggestions in real-time, right in your IDE

Categories

- Programming Languages
- Machine Learning
- Education
- Snippets

EXTENSIONS

Search Extensions in Marketplace

INSTALLED

- Dart**  
Dart language support  
✓ Dart Code
- Dev Containers**  
Open any folder or repository  
✓ Microsoft
- Docker**  
Makes it easy to create and manage containers  
✓ Microsoft
- Flutter**  
Flutter support and development  
✓ Dart Code
- GitHub Copilot**  
Your AI pair programmer  
✓ GitHub

Enable (Workspace)

**Disable**

Disable (Workspace)

Switch to Pre-Release Version

Install Another Version...

Uninstall

Copy

Copy Extension ID

Add to devcontainer.json

Extension Settings

Extension Keyboard Shortcuts

Apply Extension to all Profiles

Ignore Recommendation


Add to Workspace Recommendations

PROBLEMS

# Enabling GitHub Copilot

- ctrl-shift-x : right click copilot - Enable

Extension: GitHub Copilot X



## GitHub Copilot

v1.180.0

GitHub [github.com](#) | 15,438,095 | ★★★★★ (1020)

Your AI pair programmer

[Disable](#) [Uninstall](#) [Switch to Pre-Release Version](#)

This extension is enabled globally.

DETAILS | FEATURES | CHANGELOG | EXTENSION PACK

Followers 38k | Follow | @github | Views 16M

Try Copilot [Free trial](#)

### Your AI pair programmer

Get Code Suggestions in real-time, right in your IDE

Categories

- Programming Languages
- Machine Learning
- Education
- Snippets

EXTENSIONS

Search Extensions in Marketplace

INSTALLED

- Dart**  
Dart language support and debugger  
✓ Dart Code
- Dev Containers**  
Open any folder or repository in a container  
✓ Microsoft
- Docker**  
Makes it easy to create, manage, and deploy containerized applications  
✓ Microsoft
- Flutter**  
Flutter support and debugger  
✓ Dart Code
- GitHub Copilot**  
Your AI pair programmer  
✓ GitHub

Enable

- Enable (Workspace)
- Disable
- Disable (Workspace)
- Switch to Pre-Release Version
- Install Another Version...
- Uninstall
- Copy
- Copy Extension ID
- Add to devcontainer.json
- Extension Settings
- Extension Keyboard Shortcuts
- Apply Extension to all Profiles
- Ignore Recommendation
- Add to Workspace Recommendations



# Build auto suggest engine with Copilot

- Exercise - Use Copilot to interpret code
- Exercise - Use Copilot to autocomplete code
- Exercise - Prompt Copilot to generate code
- Exercise - Debug code with Copilot Chat
- Exercise - Test code with Copilot
- <https://learn.microsoft.com/en-us/training/modules/build-auto-suggest-engine-copilot>
- <https://github.com/MicrosoftLearning/Guided-project-Build-an-Autosuggest-Engine-with-Copilot.git>






# Gemini Code Assist

- <https://cloud.google.com/gemini/docs/codeassist/use-in-ide>
- <https://www.youtube.com/watch?v=WsXVGr0Q3C4>





# Welcome to Gemini

Get help where you need it with a conversational assistant that answers your cloud questions, reviews code snippets, and troubleshoots issues quickly.

Gemini is a work in progress and may display inaccurate information. Your prompts aren't used to train the model.

Gemini requires a Google Cloud project with the Cloud AI Companion API enabled. More details are [here](#).

Activate Gemini[Learn More](#)

Gemini is a Generative AI Feature subject to the Google Cloud [Terms of Service](#).

GEMINI: CHAT

index.html

1

Ln 1, Col 1

Spaces: 4

UTF-8

CRLF


{ } HTML

✖

✓ Prettier


🔔

# Extension




Gemini + Google Cloud Code

v2.7.1

Google Cloud  google.com | 1,145,159 | ★★★★★ (50)

Gemini + Tools for Google Cloud

Installing



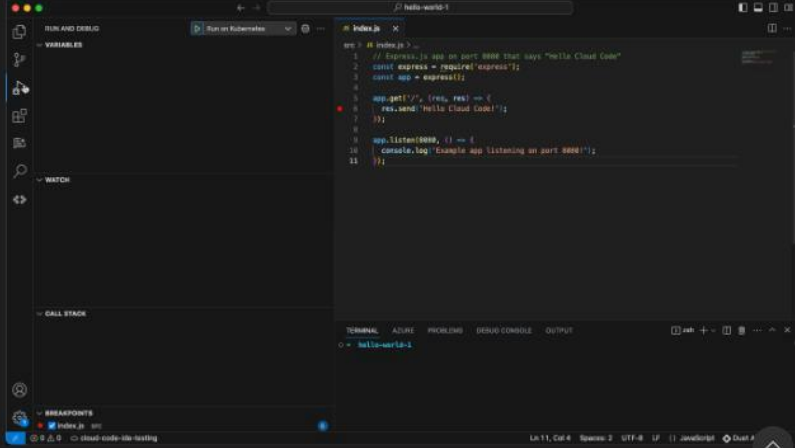
DETAILS

FEATURES

CHANGELOG

## Cloud Code for Visual Studio Code

Bring the power of Google Cloud and Gemini to help you build applications faster and easier than ever before. Gemini, your AI-powered coding assistant helps you quickly write excellent code. Cloud Code can then help you deploy your code to your favorite Google Cloud platforms with just a few clicks.



Categories

Snippets

Linters

Debuggers

Resources

Marketplace

Issues

Repository

License

Google Cloud

More Info

Published

2019-04-08, 22:05:09

Last released

2024-04-04, 04:41:04

Identifier

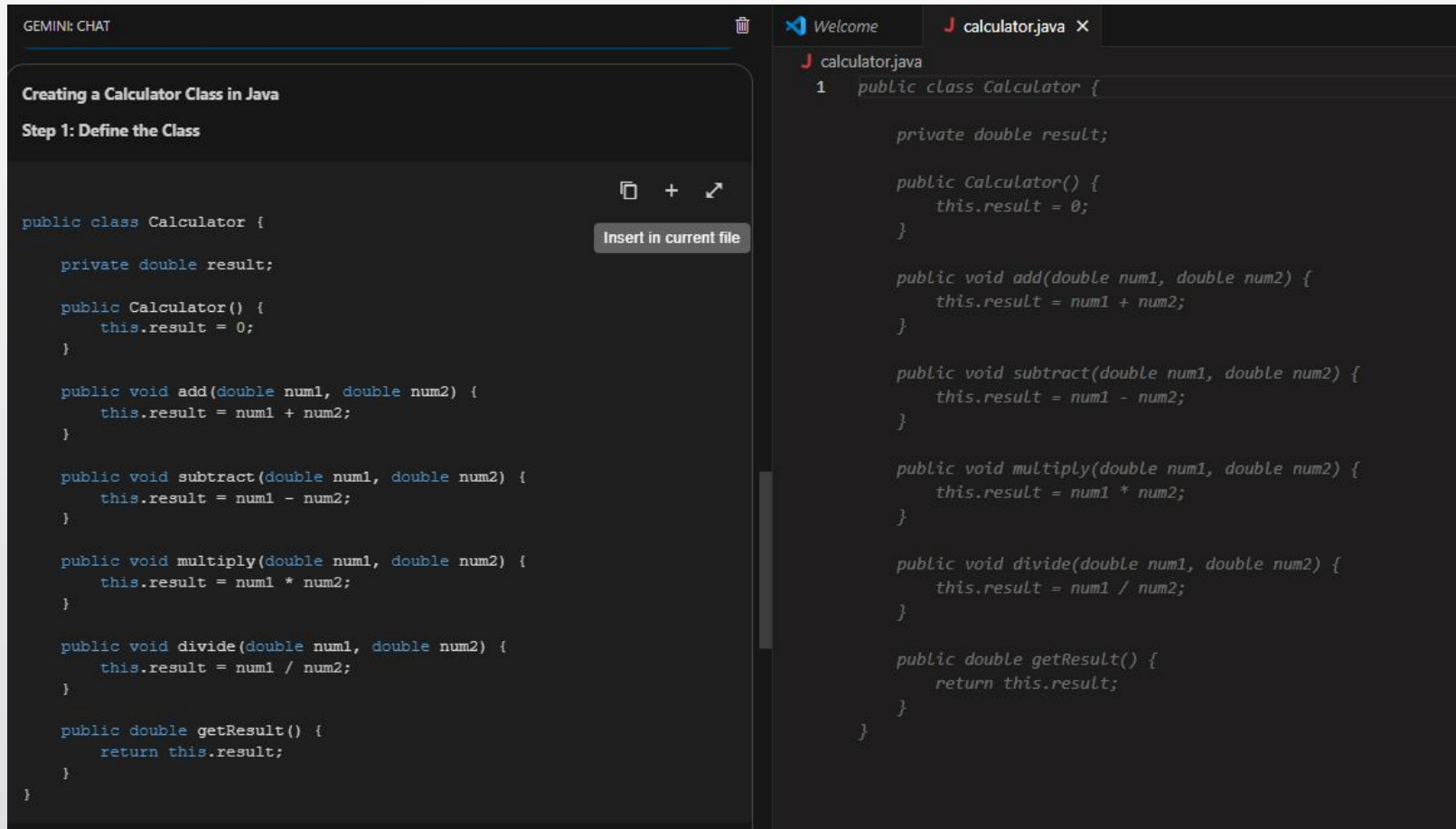
googlecloudtools.cloud

Activate Windows

Go to Settings to activate Windows.

Key Features

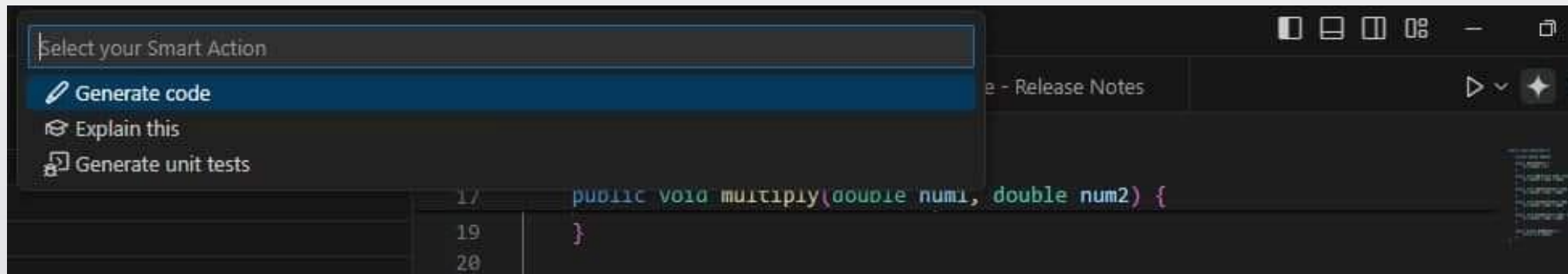
# Generate Code (Ghost Text)



The screenshot displays a development environment with two panels. The left panel, titled 'GEMINI CHAT', shows a conversation where the user asks 'Creating a Calculator Class in Java' and the assistant responds with 'Step 1: Define the Class'. Below this, the assistant provides a Java code snippet for a 'Calculator' class. The right panel shows the 'calculator.java' file in the IDE, where the generated code has been inserted. The code defines a 'Calculator' class with a private 'result' attribute and methods for addition, subtraction, multiplication, division, and retrieving the result.

```
public class Calculator {  
    private double result;  
  
    public Calculator() {  
        this.result = 0;  
    }  
  
    public void add(double num1, double num2) {  
        this.result = num1 + num2;  
    }  
  
    public void subtract(double num1, double num2) {  
        this.result = num1 - num2;  
    }  
  
    public void multiply(double num1, double num2) {  
        this.result = num1 * num2;  
    }  
  
    public void divide(double num1, double num2) {  
        this.result = num1 / num2;  
    }  
  
    public double getResult() {  
        return this.result;  
    }  
}
```

# Smart Action



[illegible]



Result

## Login

Welcome back! Please login to your account.

LOGIN

RESET



# Ctrl+Enter

```
<> login.html  < 1/3 > Accept [Tab] Accept Word [Ctrl] + [RightArrow] ...
1 //create login form
  <!DOCTYPE html>
  <html Lang="en">
    <head>
      <meta charset="UTF-8" />
      <meta http-equiv="X-UA-Compatible" content="IE=edge" />
      <meta name="viewport" content="width=device-width, initial-scale=1.0" />
      <title>Login</title>
      <style>
        .container {
          display: flex;
          flex-direction: column;
          align-items: center;
          margin: 24px;
        }
        .input-container {
          display: flex;
          flex-direction: column;
          margin: 24px;
        }
        .mdc-text-field {
          width: 300px;
        }
        .button-container {
          display: flex;
          gap: 12px;
        }
        .mdc-button {
          margin: 24px;
        }
      </style>
    </head>
  </html>
```

# DayDiff

```
1 //create daydiff function
function daydiff($date1, $date2) {
    //convert dates to timestamps
    $timestamp1 = strtotime($date1);
    $timestamp2 = strtotime($date2);
    //calculate the difference in days
    $days = ($timestamp2 - $timestamp1) / (60 * 60 * 24);
    //return the difference in days
    return $days;
}
//call daydiff function
$days = daydiff("2023-03-08", "2023-03-15");
echo "The difference in days between 2023-03-08 and 2023-03-15 is: $days";
```



# How to use a Comment to Prompt GitHub Copilot

- `// function to get the year of birth`
- `// function to get the year of birth *from the age*`



# PHP

- DayDiff
- Login Form
- Calculator Class
- PHPUnit Test
  - <https://medium.com/@naruapon/การก้าวสู่นักพัฒนามืออาชีพ-กับการทดสอบแบบหน่วยย่อย-phpunit-9-0-1-ในเชิงปฏิบัติ-44692908b764>



# Reference

- <https://github.com/features/copilot/getting-started>
- <https://nira.com/github-copilot/>
- <https://devblogs.microsoft.com/visualstudio/how-to-install-github-copilot-in-visual-studio/>
- <https://github.blog/2024-01-22-10-unexpected-ways-to-use-github-copilot>
- <https://learn.microsoft.com/en-us/training/modules/introduction-prompt-engineering-with-github-copilot/>
- <https://devblogs.microsoft.com/visualstudio/how-to-use-comments-to-prompt-github-copilot-visual-studio/>

# Reference

- <https://devblogs.microsoft.com/visualstudio/github-copilot-in-visual-studio-a-recap-of-2023/>
- <https://learn.microsoft.com/en-us/training/modules/build-auto-suggest-engine-copilot>
- Gemini for Application Developers
- <https://www.youtube.com/watch?v=WsXVGr0Q3C4>
- <https://cloud.google.com/gemini/docs/codeassist/use-in-ide>