100 XP

Examine GitHub Copilot's AI assistance features

5 minutes

GitHub Copilot Chat integrates with the Visual Studio Code user interface to provide assistance where you need it.

Here are some ways that you can access GitHub Copilot Chat features within Visual Studio Code:

- Open the Chat view for an AI assistant that can help you at any time.
- Start an inline chat conversation directly from the editor for help while you're coding.
- Run smart actions to complete certain tasks without even having to write a prompt.
- Open the Quick Chat window for a quick, interactive conversation with the AI.

Use cases for GitHub Copilot Chat

GitHub Copilot Chat offers assistance for most coding scenarios. The following sections describe some of these scenarios.

Explaining and documenting code

Copilot Chat can help explain selected code by generating natural language descriptions of the code's functionality and purpose. This can be useful if you want to understand the code's behavior or for nontechnical stakeholders who need to understand how the code works. For example, if you select a function or code block in the code editor, Copilot Chat can generate a natural language description of what the code does and how it fits into the overall system. This can include information such as the function's input and output parameters, its dependencies, and its purpose in the larger application.

By generating explanations and documentation, Copilot Chat may help you to understand the selected code, leading to improved collaboration and more effective software development.

Answering coding questions

You can ask Copilot Chat for help or clarification on specific coding problems and receive responses in natural language format or in code snippet format. This is a useful tool for programmers because it provides guidance and support for common coding tasks and challenges.

Proposing bug fixes

Copilot Chat can propose a fix for bugs in your code by suggesting code snippets and solutions based on the context of the error or issue. This is useful if you're struggling to identify the root cause of a bug or you need guidance on the best way to fix it. For example, if your code produces an error message or warning, Copilot Chat can suggest possible fixes based on the error message, the code's syntax, and the surrounding code.

Copilot Chat can suggest changes to variables, control structures, or function calls that might resolve the issue and generate code snippets that can be incorporated into the codebase. However, it's important to note that the suggested fixes may not always be optimal or complete, so you'll need to review and test the suggestions.

Generating unit test cases

Copilot Chat can help you write unit test cases by generating code snippets based on the code open in the editor or the code snippet you highlight in the editor. This helps you write test cases without spending as much time on repetitive tasks. For example, if you're writing a test case for a specific function, you can use Copilot Chat to suggest possible input parameters and expected output values based on the function's signature and body. Copilot Chat can also suggest assertions that ensure the function is working correctly, based on the code's context and semantics.

Copilot Chat can also help you write test cases for edge cases and boundary conditions that might be difficult to identify manually. For instance, Copilot Chat can suggest test cases for error handling, null values, or unexpected input types, helping you ensure your code is robust and resilient. However, it's important to note that generated test cases may not cover all possible scenarios, and manual testing and code review are still necessary to ensure the quality of the code.

Suggesting improvements to an existing codebase

Copilot Chat can also suggest potential improvements to selected code. For example, Copilot Chat can suggest improvements in the following categories:

- **Code quality**: Copilot Chat can suggest ways to improve the readability, maintainability, and performance of your code. This can include suggestions for refactoring, code simplification, and modularity.
- **Code reliability**: Copilot Chat can suggest ways to make your code more robust and reliable. This can include suggestions for error handling, input validation, and defensive programming.
- **Code performance**: Copilot Chat can suggest ways to optimize the performance of your code. This can include suggestions for algorithmic improvements, data structure optimizations, and parallelization.
- **Code security**: Copilot Chat can suggest ways to make your code more secure. This can include suggestions for input sanitization, access control, and encryption.

By suggesting improvements, Copilot Chat may help you to write better code that is more readable, reliable, performant, and secure.

How it works

GitHub Copilot Chat uses a combination of natural language processing and machine learning to understand your question and provide you with an answer. This process can be broken down into the following steps.

Input processing

The input prompt from the user is preprocessed by the Copilot Chat system and sent to a large language model to get a response based on the context and prompt. User input can take the form of code snippets or plain language. The system is only intended to respond to coding-related questions.

Language model analysis

The preprocessed prompt is then passed through the Copilot Chat language model, which is a neural network that has been trained on a large body of text data. The language model analyzes the input prompt.

Response generation

The language model generates a response based on its analysis of the input prompt and the context provided to it. This response can take the form of generated code, code suggestions, or explanations of existing code.

Output formatting

The response generated by Copilot Chat is formatted and presented to the user. Copilot Chat may use syntax highlighting, indentation, and other formatting features to add clarity to the generated response. Depending upon the type of question from the user, links to context that the model used when generating a response, such as source code files or documentation, may also be provided.

GitHub Copilot Chat is intended to provide you with the most relevant answer to your question. However, it may not always provide the answer you're looking for. Users of Copilot Chat are responsible for reviewing and validating responses generated by the system to ensure they're accurate and appropriate.

Summary

GitHub Copilot Chat integrates with the Visual Studio Code user interface to provide assistance where you need it. You can use Copilot Chat to explain and document code, answer coding questions, propose bug fixes, generate unit test cases, and suggest improvements to an existing codebase. Copilot Chat uses a combination of natural language processing and machine learning to understand your question and provide you with an answer.

Next unit: Examine GitHub Copilot Chat view features

Previous Next