About GitHub Copilot Chat

In this article

GitHub Copilot Chat can help you by providing answers to coding related questions directly within a supported IDE.

GitHub Copilot Chat is currently in public beta, and is subject to changes.

Owners of organizations or enterprises with a GitHub Copilot for Business subscription can decide whether to grant access to the GitHub Copilot Chat beta for users in their organization or enterprise.

If you have a GitHub Copilot for Individuals subscription, you now have access to the GitHub Copilot Chat beta.

About GitHub Copilot Chat

GitHub Copilot Chat is a chat interface that lets you interact with GitHub Copilot, to ask and receive answers to coding-related questions from directly within a supported IDE. The chat interface provides access to coding information and support without requiring you to navigate documentation or search online forums. Copilot Chat is currently supported in Visual Studio Code and Visual Studio. For more information about GitHub Copilot, see "About GitHub Copilot for Individuals" and "About GitHub Copilot for Business."

GitHub Copilot Chat can answer a wide range of coding-related questions on topics including syntax, programming concepts, test cases, debugging, and more. GitHub Copilot Chat is not designed to answer non-coding questions or provide general information on topics outside of coding.

GitHub Copilot Chat works by using 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 a number of steps.

Input processing

The input prompt from the user is pre-processed 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 pre-processed 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. 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.

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 are looking for. Users of Copilot Chat are responsible for reviewing and validating responses generated by the system to ensure they are accurate and appropriate. Copilot Chat is also designed to learn from your feedback and improve over time. For more information on improving the performance of GitHub Copilot Chat, see "Improving performance for GitHub Copilot Chat."

Use cases for GitHub Copilot Chat

GitHub Copilot Chat can provide coding assistance in a variety of scenarios.

Generating unit test cases

Copilot Chat can help 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 may help you write test cases without spending as much time on repetitive tasks. For example, if you are 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 is 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. For more information on generating unit test cases, see "Asking GitHub Copilot Chat questions about your code."

Explaining 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 non-technical 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 suggesting related documentation, Copilot Chat may help you to understand the selected code, leading to improved collaboration and more effective software development. However, it's important to note that the generated explanations and documentation may not always be accurate or complete, so you'll need to review, and occasionally correct, Copilot Chat's output.

Proposing code 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 can be useful if you are 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.

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 can be a useful tool for programmers, as it can provide guidance and support for common coding tasks and challenges.

Improving performance for GitHub Copilot Chat

Copilot Chat can support a wide range of practical applications like code generation, code analysis, and code fixes, each with different performance metrics and mitigation strategies. To enhance performance and address some of the the limitations of Copilot Chat, there are various measures that you can adopt. For more information on the limitations of Copilot Chat, see "Limitations of GitHub Copilot Chat."

Keep your prompts on topic

Copilot Chat is intended to address queries related to coding exclusively. Therefore, limiting the prompt to coding questions or tasks can enhance the model's output quality.

Use Copilot Chat as a tool, not a replacement

While Copilot Chat can be a powerful tool for generating code, it is important to use it as a tool rather than a replacement for human programming. You should always review and test the code generated by Copilot Chat to ensure that it meets your requirements and is free of errors or security concerns.

Use secure coding and code review practices

While Copilot Chat can generate syntactically correct code, it may not always be secure. You should always follow best practices for secure coding, such as avoiding hard-coded passwords or SQL injection vulnerabilities, as well as following code review best practices, to address Copilot Chat's limitations.

Provide feedback

If you encounter any issues or limitations with Copilot Chat, we recommend that you provide feedback through the **share feedback** link in the Copilot Chat interface of your IDE. This can help the developers to improve the tool and address any concerns or limitations.

Stay up to date

Copilot Chat is a new technology and is likely to evolve over time. You should stay up to date with any updates or changes to the tool, as well as any new security risks or best practices that may emerge. Automated extension updates are enabled by default in Visual Studio Code and Visual Studio. If you have automatic updates enabled, Copilot Chat will automatically update to the latest version when you open your IDE. For more information on automatic updates in your IDE, see the Visual Studio Code documentation and the Visual Studio documentation.

Limitations of GitHub Copilot Chat

Depending on factors such as your codebase and input data, you may experience different levels of performance when using Copilot Chat. The following information is designed to help you understand system limitations and key concepts about performance as they apply to Copilot Chat.

Limited scope

Copilot Chat has been trained on a large body of code but still has a limited scope and may not be able to handle more complex code structures or obscure programming languages. For each language, the quality of suggestions you receive may depend on the volume and diversity of training data for that language. For example, JavaScript is well-represented in public repositories and is one of GitHub Copilot's best supported languages. Languages with less representation in public repositories may be more challenging for Copilot Chat to provide assistance with. Additionally, Copilot Chat can only suggest code based on the context of the code being written, so it may not be able to identify larger design or architectural issues.

Potential biases

Copilot's training data is drawn from existing code repositories, which may contain biases and errors that can be perpetuated by the tool. Additionally, Copilot Chat may be biased towards certain programming languages or coding styles, which can lead to suboptimal or incomplete code suggestions.

Security risks

Copilot Chat generates code based on the context of the code being written, which can potentially expose sensitive information or vulnerabilities if not used carefully. You should be careful when using Copilot Chat to generate code for security-sensitive applications and always review and test the generated code thoroughly.

Matches with public code

Copilot Chat is capable of generating new code, which it does in a probabilistic way. While the probability that it may produce code that matches code in the training set is low, a Copilot Chat suggestion may contain some code snippets that match code in the training set. Copilot Chat utilizes filters that block matches with public code on GitHub repositories, but you should always take the same precautions as you would with any code you write that uses material you did not independently originate, including precautions to ensure its suitability. These include rigorous testing, IP scanning, and checking for security vulnerabilities. You should make sure your IDE or editor does not automatically compile or run generated code before you review it.

Inaccurate code

One of the limitations of Copilot Chat is that it may generate code that appears to be valid but may not actually be semantically or syntactically correct or may not accurately reflect the intent of the developer. To mitigate the risk of inaccurate code, you should carefully review and test the generated code, particularly when dealing with critical or sensitive applications. You should also ensure that the generated code adheres to best practices and design patterns and fits within the overall architecture and style of the codebase.

Inaccurate responses to non-coding topics

Copilot Chat is not designed to answer non-coding questions, and therefore its responses may not always be accurate or helpful in these contexts. If a user asks Copilot Chat a non-coding question, it may generate an answer that is irrelevant or nonsensical, or it may simply indicate that it is unable to provide a useful response.

Next steps

- Using GitHub Copilot Chat

Further reading

- "GitHub Copilot Pre-release License Terms"
- GitHub Copilot Trust Center