# GitHub Copilot data

4 minutes

In this unit, we'll cover how GitHub Copilot handles data for different environments, features and configurations.

## Data handling for GitHub Copilot code suggestions

GitHub Copilot in the code editor does not retain any prompts like code or other context used for the purposes of providing suggestions to train the foundational models. It discards the prompts once a suggestion is returned.

GitHub Copilot Individual subscribers can opt-out of sharing their prompts with GitHub which will otherwise be used to finetune GitHub's foundational model.

## Data handling for GitHub Copilot chat

GitHub Copilot Chat operates as an interactive platform, allowing developers to engage in conversational interactions with the AI assistant to receive coding assistance. Here are the steps that it carries out which might be distinct from other features like code completion:

- **Formatting**: Copilot meticulously formats the generated response for optimal presentation within the chat interface. It highlights code snippets to improve readability and may include options for direct integration into your code. Copilot showcases the formatted response in the Copilot Chat window within the IDE, allowing you to easily review and interact with the provided information.
- **User engagement**: You can actively engage with the response by asking follow-up questions, requesting clarifications, or providing additional input. The chat interface maintains a conversation history to facilitate contextual understanding in subsequent interactions.
- **Data retention**: For Copilot Chat used outside the code editor, GitHub typically retains prompts, suggestions, and supporting context for 28 days. Specific retention policies for Copilot Chat within the code editor may vary.

The same goes for CLI, Mobile, and GitHub Copilot Chat on GitHub.com.

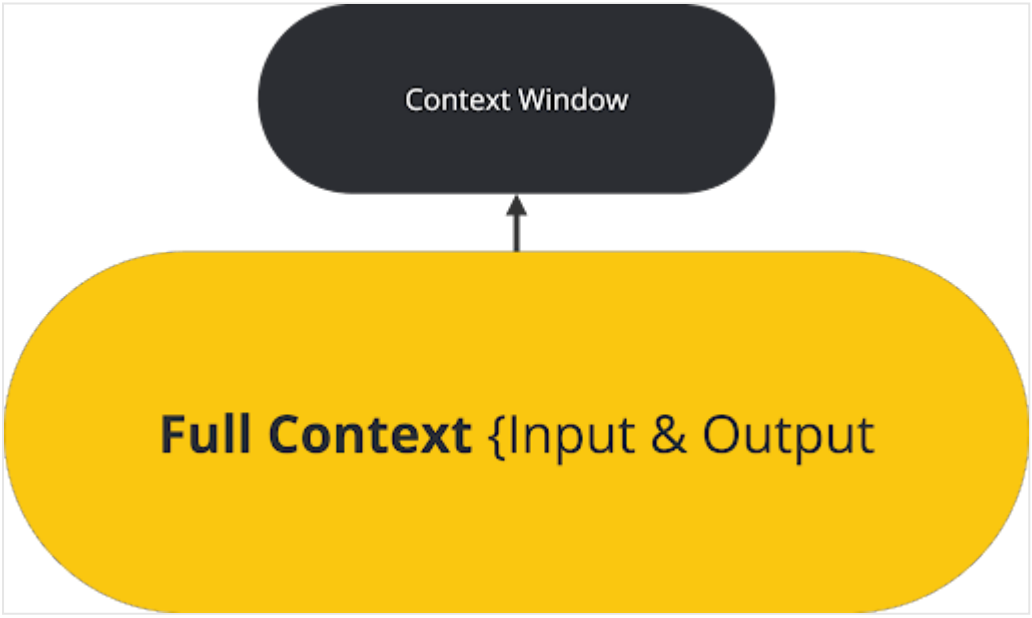
# Prompt types supported by GitHub Copilot Chat

GitHub Copilot Chat processes a wide range of coding-related prompts, demonstrating its versatility as a conversational coding assistant. Here are some common input types:

- **Direct Questions**: You can ask specific questions about coding concepts, libraries, or troubleshooting issues. For example, "How do I implement a quick sort algorithm in Python?" or "Why is my React component not rendering?"
- **Code-Related Requests**: You can request code generation, modification, or explanation. Examples include "Write a function to calculate factorial," "Fix this error in my code," or "Explain this code snippet."
- **Open-Ended Queries**: You can explore coding concepts or seek general guidance by asking open-ended questions like "What are the best practices for writing clean code?" or "How can I improve the performance of my Python application?"
- **Contextual Prompts**: You can provide code snippets or describe specific coding scenarios to seek tailored assistance. For instance, "Here's a part of my code, can you suggest improvements?" or "I'm building a web application, can you help me with the authentication flow?"

Copilot Chat's ability to process diverse input types enhances its utility as a comprehensive coding companion.

# Limited context windows




While GitHub Copilot Chat excels at understanding and responding to prompts, it's essential to acknowledge the limitation of context windows. This refers to the amount of surrounding code and text the model can process simultaneously to generate suggestions. GitHub Copilot's context window typically ranges from approximately 200-500 lines of code or up to a

few thousand tokens. This limitation can vary depending on the specific implementation and version of Copilot being used.

Copilot Chat currently operates with a context window of 4k tokens, providing a broader scope for understanding and responding to user queries compared to the standard Copilot.

Despite these advancements, you should be mindful of context window limitations when crafting prompts. Breaking down complex problems into smaller, more focused queries or providing relevant code snippets can significantly enhance the model's ability to provide accurate and helpful responses.

---

## Next unit: GitHub Copilot Large Language Models (LLMs)

Previous Next