

25

# What's New for Visual Studio Code

CMake Improvements and  
GitHub Copilot Agents

ALEXANDRA KEMPER



**Cppcon**  
The C++ Conference

20  
25



September 13 - 19

# Microsoft @ CppCon

	14:00	<b>What's new in Visual Studio for C++ Developers in 2025</b>	Augustin Popa & David Li
Tues.	14:00	Back to Basics: Code Review	Chandranath Bhattacharyya & Kathleen Baker
	14:00	LLMs in the Trenches: Boosting System Programming with AI	Ion Todirel
	15:15	C++ Performance Tips: Cutting Down on Unnecessary Objects	Kathleen Baker & Prithvi Okade
Wed.	15:50	Connecting C++ Tools to AI Agents Using the Model Context Protocol	Ben McMoran
		Welcome to v1.0 of the meta::[[verse]]!	Inbal Levi
	14:00	MSVC C++ Dynamic Debugging: How We Enabled Full Debuggability of Optimized Code	Eric Brumer
Thurs.	16:45	It's Dangerous to Go Alone: A Game Developer Tutorial	Michael Price
	9:00	Reflection-based JSON in C++ at Gigabytes per Second	Daniel Lemire & Francisco Geiman Thiesen
Fri	13:30	Duck-Tape Chronicles: Rust/C++ Interop	Victor Ciura

# Agenda

---



1. Faster Performance for C/C++ Extension



2. C++ Tooling updates

*Updating the build scripts in the VCMI repo*



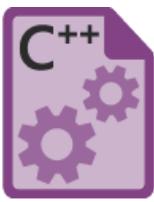
3. AI Agents

*Building a C++ proof of concept from scratch*



4. Other VS Code updates

# Extensions for C++ Development in VS Code



**Makefile Tools** pre-release Preview

Microsoft [microsoft.com](#) | ⚡ 7,907,736 | ★★★★☆ (39)

Provide makefile support in VS Code: C/C++ IntelliSense, build, debug/run.

[Disable](#) | [Uninstall](#) | [Switch to Release Version](#)  Auto Update 



**C/C++** pre-release

Microsoft [microsoft.com](#) | ⚡ 86,268,727 | ★★★★★ (588)

C/C++ IntelliSense, debugging, and code browsing.

[Disable](#) | [Uninstall](#) | [Switch to Release Version](#)  Auto Update 



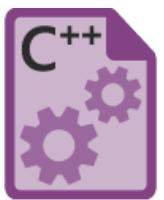
**CMake Tools** pre-release

Microsoft [microsoft.com](#) | ⚡ 49,040,256 | ★★★★★ (74)

Extended CMake support in Visual Studio Code

[Disable](#) | [Uninstall](#) | [Switch to Release Version](#)  Auto Update 

# Extensions for C++ Development in VS Code



## Makefile Tools pre-release Preview

Microsoft [microsoft.com](#) | ⚡ 7,907,736 | ★★★★☆(39)

Provide makefile support in VS Code: C/C++ IntelliSense, build, debug/run.

[Disable](#) | [Uninstall](#) | [Switch to Release Version](#)  Auto Update



## C/C++ pre-release

Microsoft [microsoft.com](#) | ⚡ 86,268,727 | ★★★★☆(588)

C/C++ IntelliSense, debugging, and code browsing.

[Disable](#) | [Uninstall](#) | [Switch to Release Version](#)  Auto Update



## CMake Tools pre-release

Microsoft [microsoft.com](#) | ⚡ 49,040,256 | ★★★★★(74)

Extended CMake support in Visual Studio Code

[Disable](#) | [Uninstall](#) | [Switch to Release Version](#)  Auto Update



## GitHub Copilot Chat pre-release

GitHub [github.com](#) | ⚡ 38,524,558 | ★★★★★(161)

AI chat features powered by Copilot

[Disable](#) | [Uninstall](#) |  Auto Update



## GitHub Copilot pre-release

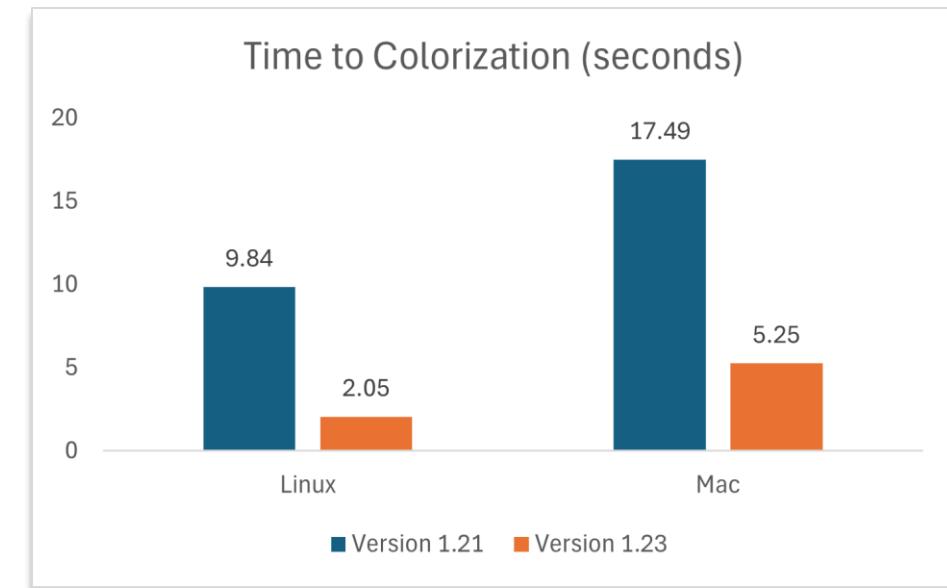
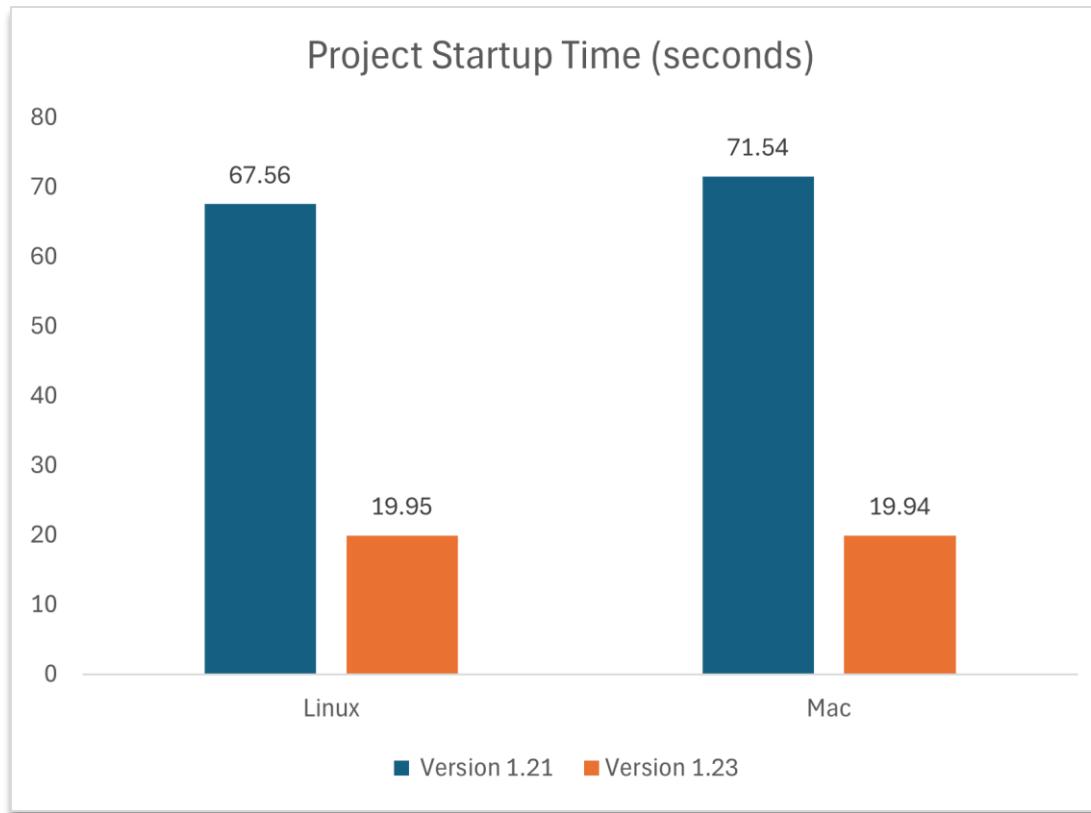
GitHub [github.com](#) | ⚡ 48,044,345 | ★★★★★(994)

Your AI pair programmer

[Disable](#) | [Uninstall](#) | [Switch to Release Version](#)  Auto Update

# Performance Gains for large C++ codebases

## Faster Colorization & Project Startup



*But how?*

- Configuration caching
- Better `compile_commands.json` support, custom config handling, file discovery...

Learn More at [aka.ms/cpptools-perf](http://aka.ms/cpptools-perf)

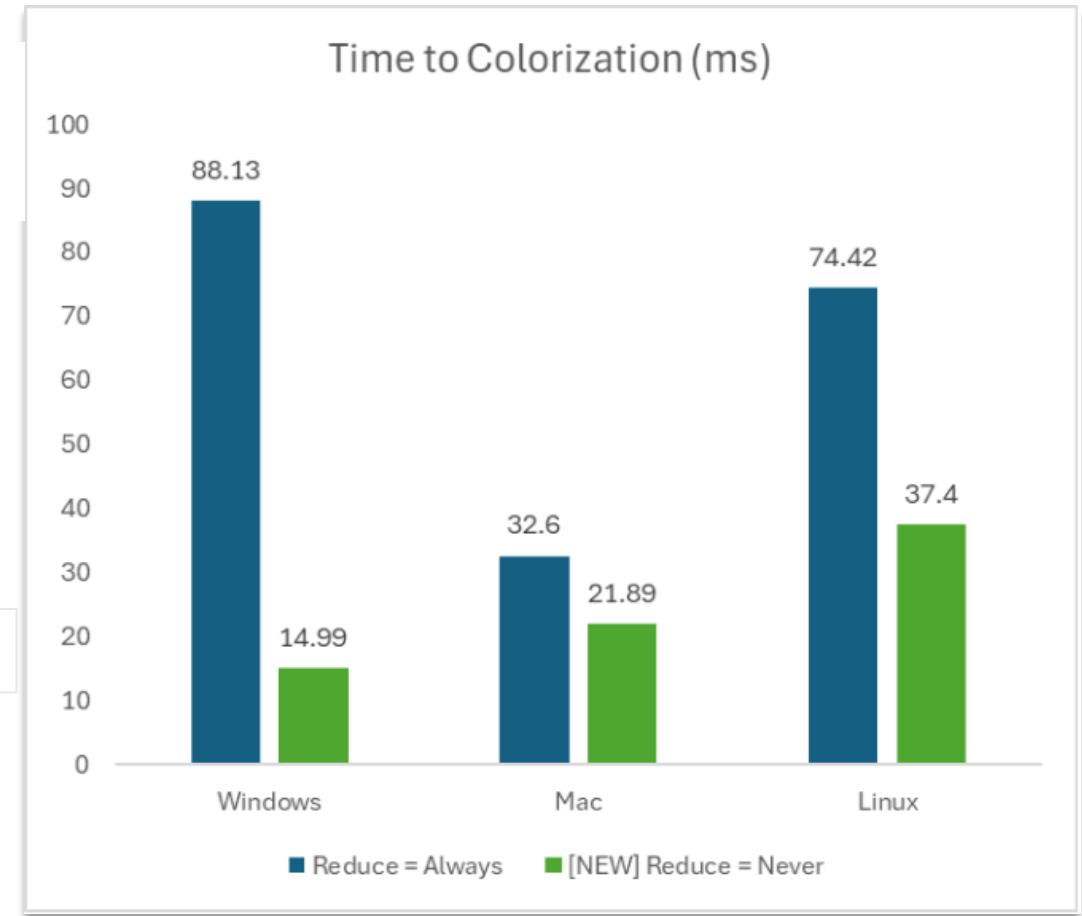
# Performance: Customize recursive #includes

Default:

```
"includePath": [  
    "${workspaceFolder}/**"  
,
```

GameEngine.cpp

```
#include "render/Canvas.h"  
#include "render/Colors.h"  
#include "render/IFont.h"  
#include "render/EFont.h"  
#include "renderSDL/ScreenHandler.h"  
#include "renderSDL/RenderHandler.h"  
  
#include ".../gui/TextAlignment.h"  
#include ".../lib/Rect.h"  
#include ".../lib/Color.h"
```



Learn More at [aka.ms/cpptools-1.25](http://aka.ms/cpptools-1.25)

# Demo: Updating your build process with the CMake Tools Extension

[vcmi/vcmi: Open-source engine for Heroes of Might and Magic III](#)

# Backup Demo Recording

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** On the left, it shows a project structure with a folder named "vcmi". Inside "vcmi", there are subfolders ".vscode", "build", and "vcmi". The "vcmi" folder is currently selected.
- Editor:** The main editor area displays the content of the "CMakePresets.json" file. The code defines three configuration presets: "release-binary-dir", "build-with-conan", and "base-release".

```
1  {
2      "version": 2,
3      "configurePreset": [
4          {
5              "name": "release-binary-dir",
6              "hidden": true,
7              "binaryDir": "${sourceDir}/out/build/${presetName}"
8          },
9          {
10             "name": "build-with-conan",
11             "hidden": true,
12             "cacheVariables": {
13                 "CMAKE_TOOLCHAIN_FILE": "${sourceDir}/conan-generated/conan_toolchain.cmake",
14                 "FORCE_BUNDLED_MINIZIP": "OFF"
15             }
16         },
17         {
18             "name": "base-release",
19             "inherits": "release-binary-dir",
20             "hidden": true,
21             "cacheVariables": {
22                 "CMAKE_INSTALL_PREFIX": "${sourceDir}/out/install/${presetName}",
23                 "PACKAGE_FILE_NAME": "$env{VCMI_PACKAGE_FILE_NAME}",
24                 "PACKAGE_NAME_SUFFIX": "$env{VCMI_PACKAGE_NAME_SUFFIX}",
25                 "CMAKE_BUILD_TYPE": "RelWithDebInfo",
26                 "ENABLE_TEST": "OFF",
27                 "ENABLE_STRICT_COMPILATION": "ON",
28                 "ENABLE_GOLMASTER": "$env{VCMI_PACKAGE_GOLMASTER}",
29                 "ENABLE_PCH": "OFF"
30             }
31         }
32     ]
33 }
```
- Bottom Status Bar:** Shows the file path "vcmi CMakePresets.json", the status "Configuring project: Saving open files", the author "Konstantin (3 years ago)", the line number "Ln 31, Col 11", and the file encoding "UTF-8 CRLF".

# Recap: Modernizing your Cmake Scripts

## CMake Language Services – now provided by CMake Tools extension

```
set(CMAKE_VS_GLOBALS
```

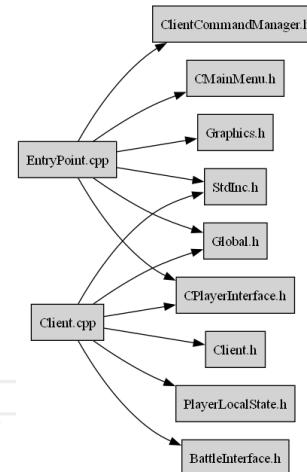
Add compile definitions to a target.

```
target_compile_definitions(<target> <INTERFACE|PUBLIC|PRIVATE> [items1...]
[<INTERFACE|PUBLIC|PRIVATE> [items2...] ...])
target_compile_definitions(foo PUBLIC FOO) target_compile_definitions(foo PUBLIC -DFOO)
target_compile_definitions(foo PUBLIC "" FOO) target_compile_definitions(foo PUBLIC -D FOO)
target_compile_definitions(foo PUBLIC FOO=1)

target_compile_definitions(VCMI PRIVATE
```

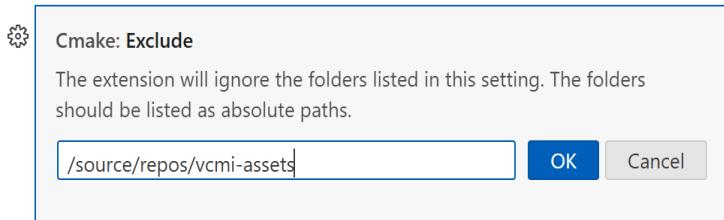
## Support through CMake Presets v10

```
{} CMakePresets.json M •
{} CMakePresets.json > [ ] configurePresets
1 {
2   "version": 10,
3   "configurePresets": [
4     {
5       "name": "macos-ninja-release",
6       "displayName": "Ninja release",
7       "description": "VCMI MacOS Ninja",
8       "inherits": "default-release",
9       "$comment": "MacOS build with Ninja for better performance"
```

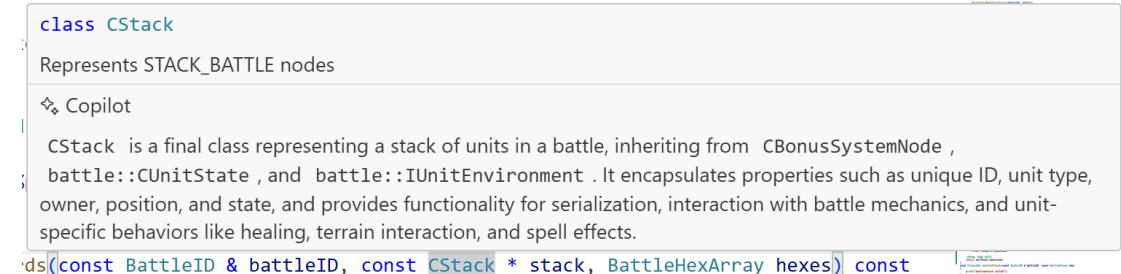


# Recap: Improvements to code navigation + understanding

# Multi-root configurability for CMake



# Copilot Hover



## Updates to C++ Completions: more **context-aware**

- Next-Edit Suggestions predict where and what you need to update next

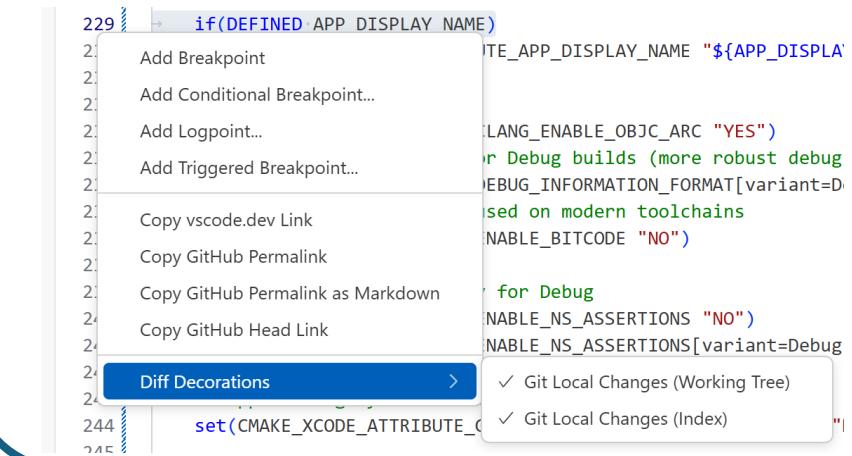
GitHub > Copilot > Next Edit Suggestions: Enabled

Whether to enable next edit suggestions (NES).

NES can propose a next edit based on your recent changes. [Learn more](#) about next edit suggestions.

- New, updated models trained on more C++
  - Contextual Improvements tailored to C++ coding structure

## View your Git staged commits from editor



# But what about building from scratch?

CHAT

What steps do I need to take to build the game snake?

Here are the steps to build the game Snake as a proof of concept in your CMake-based project (using C++ and SDL2, which you already have as a dependency):

- 1. Create a New Source File**
  - Add a new file, e.g., `snake_game.cpp`, in your project (e.g., in the `vcmi` or `client` directory).
- 2. Implement the Snake Game Logic**
  - In `snake_game.cpp`, implement a minimal Snake game using SDL2 for windowing, input, and rendering.
- 3. Add a New Executable Target in CMake**
  - In your `CMakeLists.txt`, add:

```
add_executable(snake_game snake_game.cpp)
target_link_libraries(snake_game PRIVATE SDL2::SDL2)
```
- 4. Build the Project**
  - Use your existing CMake presets or build system:
    - Configure:  
`cmake -S . -B build -G "Ninja"`
    - Build:  
`cmake --build build --target snake_game`
- 5. Run the Game**
  - Run the resulting executable (e.g., `build/snake_game.exe` on Windows).

**Summary of steps:**

- Add `snake_game.cpp` with game logic.

Add context (#), extensions (@), commands (/)

Ask GPT-4.1

Add Context... CMakePresets.json +

# Codebase wide changes, powered by Agents

Intelligent systems that

- Accept natural language instead of step-by-step instructions
- Automate complex workflows
- Can invoke tools & commands (compilers, debuggers, etc.)
- Can edit across multiple files



*“Add a thread-safe logging utility to this C++ project”*

Demo: Building from Scratch with custom  
instructions

# Backup recording

The screenshot shows the Microsoft Visual Studio Code interface with a large 'X' watermark displayed across the center of the workspace. The interface includes:

- File Bar:** File, Edit, Selection, View, Go, ...
- Search Bar:** DependencyVisualizer
- Icons:** Explorer, GitHub, Search, Connect, Terminal, Split, Find, Log, Snippet, Help, ...
- EXPLORER:** DEPENDENCYVISUALIZER (containing .github, prompts, copilot-instructions.md, build)
- CHAT:** Chat icon, AI response icon, "Build with agent mode." message, AI responses may be inaccurate, disable telemetry link.
- PROBLEMS:** Problems icon, Filter (e.g. text, author) input, There are no comments in this workspace yet.
- OUTLINE:** Outline icon, Timeline icon.
- Bottom Bar:** Visualizer, Build, Live Share, and other standard VS Code icons.

# Copilot Coding Agent



Async & in the cloud

- Integrated into GitHub
- Explores repo, writes code, runs tests, opens a pull request for review
- Works in the background

## Example:

- Fix a bug in the codebase & create a relevant PR

# Agent Mode in VS Code



Real-time & in VS Code

- Local & interactive for specific task that iterates until successful
- Agent edits your code & invokes tools
- Immediate feedback & control

## Examples:

- Plan and implement new features
- Refactor parts of your codebase

Learn More at [aka.ms/agent-mode](https://aka.ms/agent-mode) & [aka.ms/copilot-coding-agent](https://aka.ms/copilot-coding-agent)

# Recap from demo: Customize Copilot

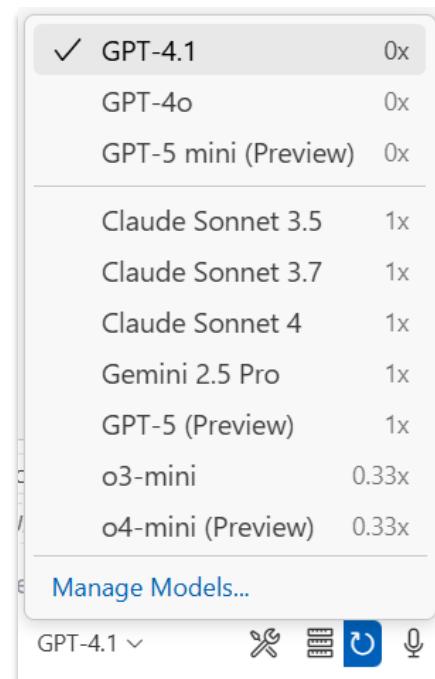
## Custom Instructions

Specify coding standards based on your preferences per workspace

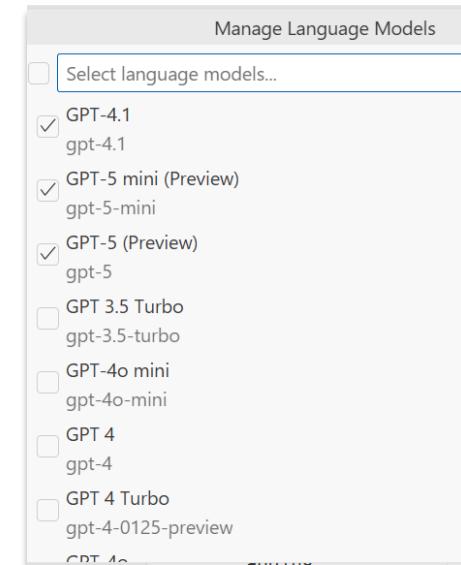
```
GITHUB_COPILOT_INSTRUCTIONS.md M X
GITHUB_COPILOT_INSTRUCTIONS.md > # C++ Coding Preferences
1
2 # C++ Coding Preferences
3
4 **General**
5 - Target **C++23** where available; otherwise
**C++20**. Prefer standard library features
over third-party deps. Avoid adding
dependencies unless I ask.
```

## Model Picker

Choose between a variety of different Models



Bring Your Own Key  
Add additional models beyond built-in providers



## Prompt Files

define reusable prompts for common development tasks

```
.github > prompts > generate-diagram.prompt.md
1 ---
2 mode: 'agent'
3 description: 'Generate a visualizer for file level dependencies'
4 ---
5
6 Create a C++20 project which contains multiple files using CMake
that analyzes a C++ codebase and visualizes file-level dependencies
```

Learn More at [aka.ms/customize-copilot](https://aka.ms/customize-copilot)

# Extensibility for Agent: Tools + MCP

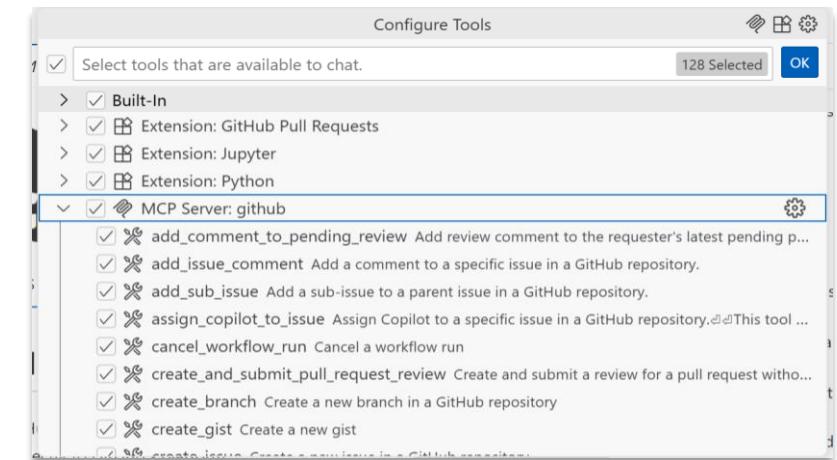
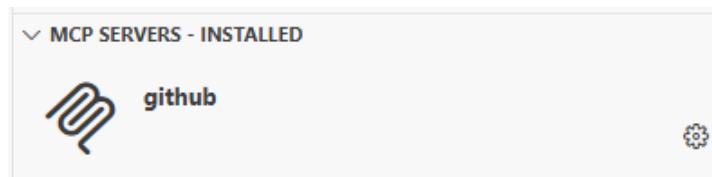
**Tools** extend the agent's capabilities beyond basic code generation, enabling it to interact with your environment, run commands, and more...

- Invoke manually and/or automatically by agent
- Agent supports tools provided by VS Code ecosystem (built-in and extensions) and MCP server tools
- Can enable/disable

---

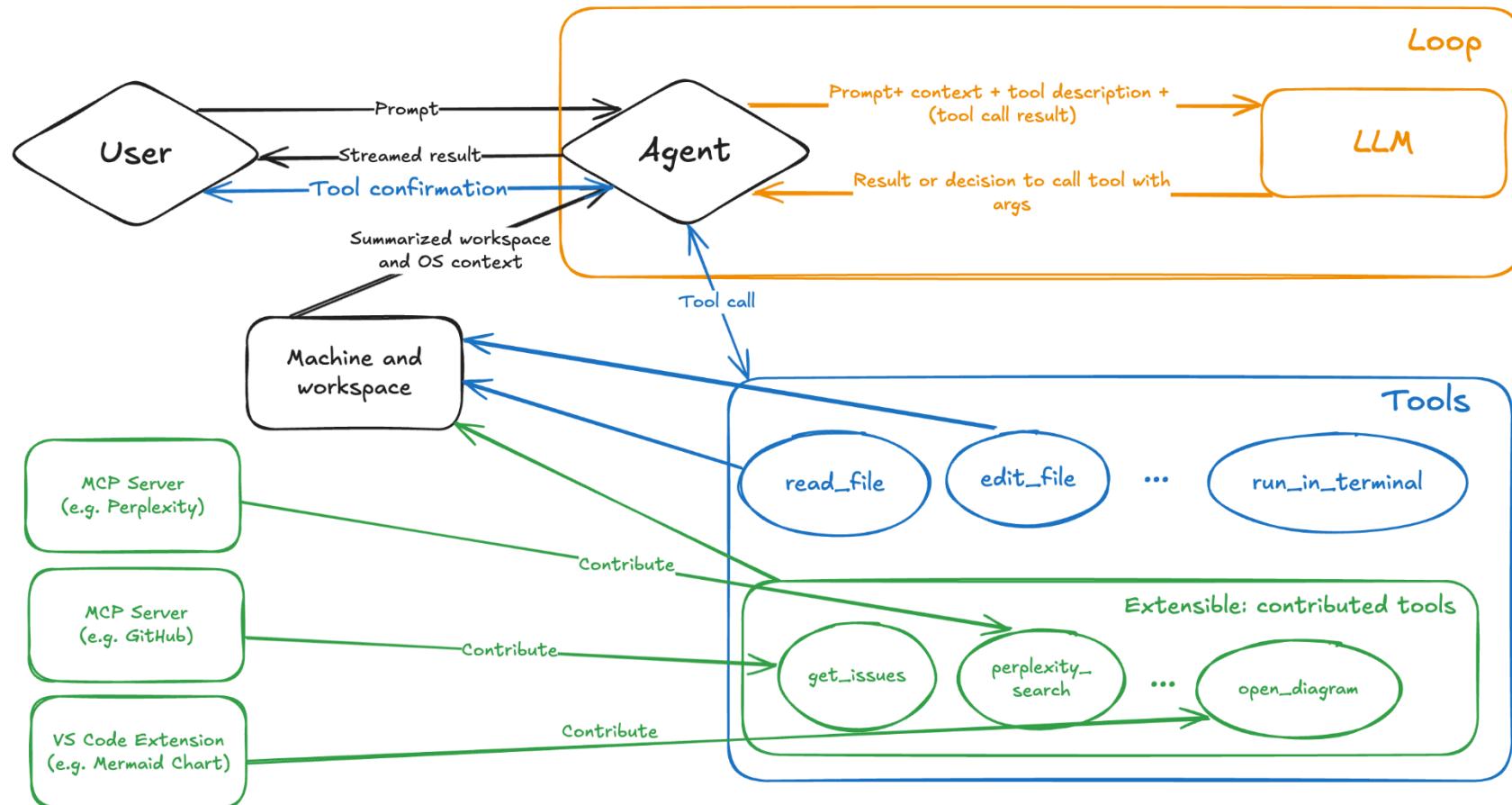
**Model Context Protocol (MCP)** enables GitHub Copilot to connect with wide range of external tools and data sources

- **MCP tools** are **server-based extensions** that Copilot can invoke in Agent Mode
- Configure enterprise-wide permission policies



Learn More at the MCP talk on Wed @15:50

# Extensibility for Agents: Tools + MCP

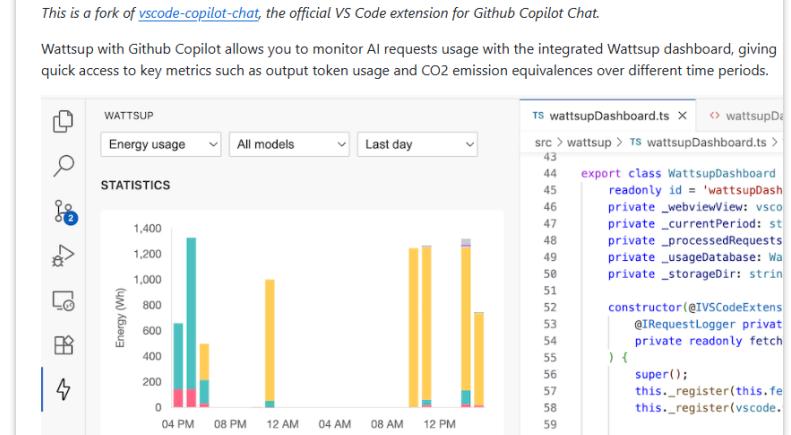


# GitHub Copilot: Free + Open-Source

- Copilot Chat Extension is now open source
  - Audit, extend, and customize chat

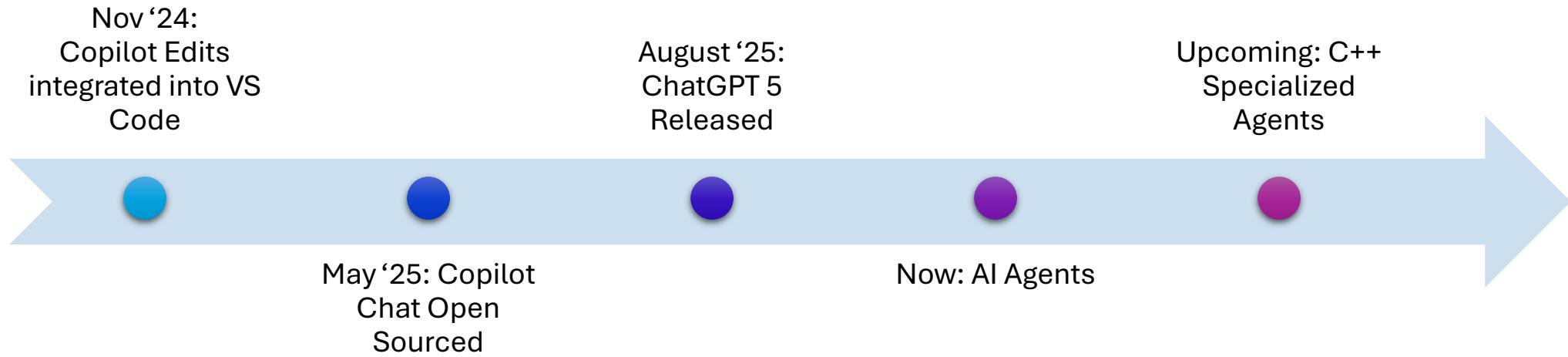


## Wattsup with Github Copilot



Want to try copilot? GitHub Copilot Free Tier now available, sign up at <https://github.com/features/copilot>

# Where we are now?



# Questions?



@AlexandraKemperMS  
@VisualC



visualcpp@microsoft.com