Overview

Introduction

Copy page

Get started with the Agent Client Protocol (ACP)

The Agent Client Protocol standardizes communication between code editors (IDEs, text-editors, etc.) and coding agents (programs that use generative AI to autonomously modify code).

Why ACP?

AI coding agents and editors are tightly coupled but interoperability isn't the default. Each editor must build custom integrations for every agent they want to support, and agents must implement editor-specific APIs to reach users. This creates several problems:

- Integration overhead: Every new agent-editor combination requires custom work
- Limited compatibility: Agents work with only a subset of available editors
- Developer lock-in: Choosing an agent often means accepting their available interfaces

ACP solves this by providing a standardized protocol for agent-editor communication, similar to how the **Language Server Protocol (LSP)** standardized language server integration.

Agents that implement ACP work with any compatible editor. Editors that support ACP gain access to the entire ecosystem of ACP-compatible agents. This decoupling allows both sides to innovate independently while giving developers the freedom to choose the best tools for their workflow.

Overview

ACP assumes that the user is primarily in their editor, and wants to reach out and use agents to assist them with specific tasks.

Agents run as sub-processes of the code editor, and communicate using JSON-RPC over stdio. The protocol re-uses the JSON representations used in MCP where possible, but includes custom types for useful agentic coding UX elements, like displaying diffs.

The default format for user-readable text is Markdown, which allows enough flexibility to represent rich formatting without requiring that the code editor is capable of rendering HTML.

Architecture Next ›