MCP in Claude Code

3 MIN · YOUTUBE · HTTPS://WWW.YOUTUBE.COM/WATCH?V=KKBFMWKDZDO
https://www.youtube.com/watch?v=kkBFmwkDzdo

SUMMARY

Model Contact Protocol (MCP) is an open standard that enables Claude code to connect with external tools and data sources, enhancing its ability to understand and respond to queries. By managing context effectively, users can optimize their interactions with Claude and ensure efficient use of resources.

- *MCP allows Claude to utilize external tools for improved query understanding.*
- *Users can add MCP servers using the Claude MCP add command, with two types: HTTP (remote services) and STDIO (local processes).*
- *MCP servers can be scoped locally, per user, or project-wide using a .mcp.json file for consistency across teams.*
- *It's important to manage connected servers to avoid excessive context usage, which can hinder performance.*
- *CLI tools are more context-efficient than MCP servers, as they do not add persistent definitions to the context window.*
- *If MCP tools exceed 10% of the context window, Claude will switch to tool search mode, which may be less effective.*
- *Regularly check connected servers with the /mcp command to disable unused tools and optimize context usage.*

Model contact protocol is an open standard that lets Claude code connect to external tools and data sources. >> [music] >> When you ask a question, Claude will automatically understand when it should use those tools to better understand your query. Context is one of the most important parts when working with Claude code. A lot of your context lives elsewhere like your databases, your productivity apps, or [music] in public repositories. This is where MCP comes [music] in.

First, it's important to understand the concept of tools when talking about Agentic AI. Tools give agents like Claude code the ability to perform actions in order for them to better complete their tasks. This is different from other AI where you just get an output back directly in text usually. For example, if your team is using Linear as our project management software, you can add a Linear MCP server to bring in the details of your specific issues.

If you want to get up-to-date documentation of a dependency that you're working with, then the Context 7 MCP server will provide Claude code with that. There are also hundreds of different connectors at claude.com/connectors. You can add MCP servers with the Claude MCP add command. There are two main types. HTTP servers are for remote services. These are hosted by the service provider and connect over the network. STDIO servers are for local processes that run on your machine.

You can manage your servers with the /mcp inside a Claude code session to see what's connected, the status, and

disable servers that you don't want to use. MCP servers can be scoped in three different ways. One, local means it's only available in the current project for you. Two, the user, which means it's available across all your projects. And three, project scope uses a .mcp.json file that you check into your version control, so anyone working on the code base gets the exact same servers automatically.

Now, one thing to be aware of is that MCP servers add tool definitions to your context window, even when you're not using them. So, if you have a lot of servers configured, this eats into your available context. Run the {slash} MCP command to see what's connected and disable anything that you're not actively using or don't think that you're going to use. If a tool has a CLI equivalent like GH for GitHub or AWS for AWS, the CLI is more context efficient because it doesn't add persistent tool definitions.

You also might benefit from using a skill in this scenario. A skill has a name and a description that is loaded into context. Similar to MCP, when Cloud thinks it needs to use that skill, it then decides to load it into the context window, which is where you could put the command line interface tools. If your MCP tools exceed 10% of your context window, Cloud code will automatically switch to tool search mode, which will discover the right tools on demand, but this might not work as well since it's just not in the context.

>> [music] >> Now, a quick recap. MCP connects Cloud code to your external tools and data sources. >> [music] >> Add servers with Cloud MCP add, scope them to your project with .mcp.json so that your team gets them automatically, and keep an eye on the context usage by disabling servers [music] that you're not actively using.