Connecting Claude to MCP tools

Use Claude.ai in your browser to connect to one of these workshop servers.

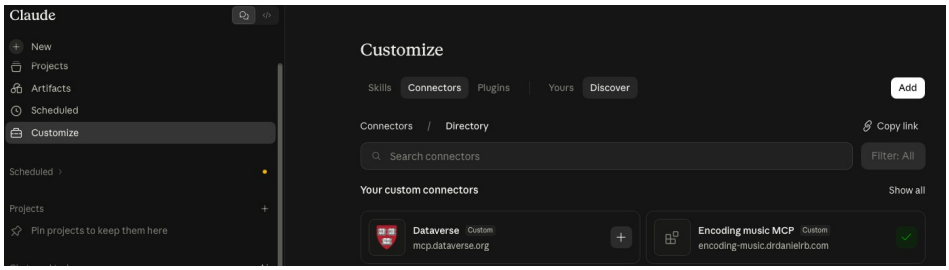
Endpoint	Remote MCP server URL and purpose
Draw.io	https://mcp.draw.io/mcp Create editable diagrams from text descriptions. Docs https://github.com/jgraph/drawio-mcp
PubMed	https://pubmed.mcp.claude.com/mcp Search biomedical literature and retrieve article details. Docs https://claude.com/resources/tutorials/using-the-pubmed-connector-in-claude
World Bank Data360	https://maimcpext.worldbank.org/ext/data360/mcp Retrieve World Bank development indicators and statistics. Docs https://worldbank.github.io/data360-mcp/
Encoding Music	https://encoding-music.drdanielrb.com/mcp Analyse encoded music scores and display notation or audio. Docs https://github.com/unimelbmdap/encoding-music-mcp

Add your connector

> **Before you start** Sign in at https://claude.ai. Free accounts can add one custom connector. For Team or Enterprise, an owner must add the connector first. If you get stuck at any point, ask the facilitator or follow along with another participant.

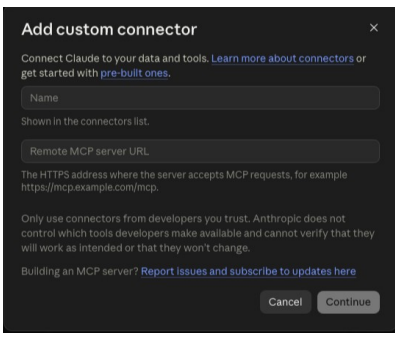
1. **Open Connectors.** Choose **Customize** › **Connectors** in the left sidebar, or go straight to https://claude.ai/customize/connectors.

2. **Add a connector.** Click **Add** at the top-right to begin adding your connector.



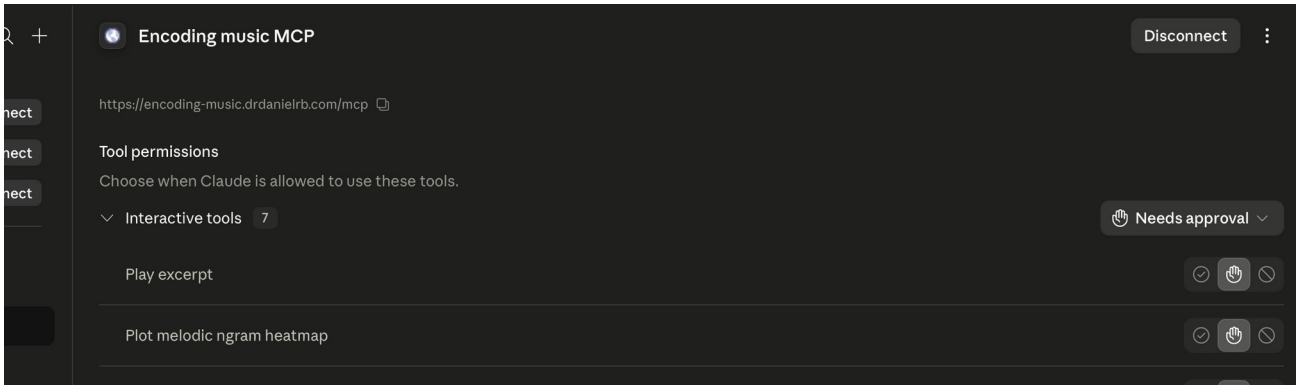
Customize › Connectors. The Add button is at the top-right.

3. **Enter the details.** Enter the server's name in **Name** and paste its matching URL from the table into **Remote MCP server URL**. Click **Continue**, then **Add** on the authentication screen that follows. These workshop servers are open, so leave **Authentication** on the detected setting.



Enter your chosen server's name and URL, then click Continue.

4. **Review permissions.** The connector's page opens once it is added. Review its tools and their permission settings.



The connector page shows its tools and their permission settings.

Leave tools on **Needs approval** while you are learning, so you see each call before it runs. Check the tool name and its inputs before you approve.

5. **Enable it in a chat.** Click **New** in the left sidebar to start a chat. Click **+** at the lower-left of the message box, choose **Connectors** and switch your connector on for this conversation.

6. **Check it works.** Whichever server you connected, ask three questions in sequence: confirm the connection, list the tools, then ask for something that needs one.

THE PATTERN: ANY SERVER

a) Do you have access to the [server name] MCP server?

b) What tools does it have available?

c) [Create a request that could use one or more of the available tools.]

WORKED EXAMPLE: DRAW.IO

a) *Do you have access to the DrawIO MCP server?*

b) *What tools does it have available?*

c) *Make me a diagram that shows...*

Look for an actual tool call in the reply and approve it when prompted.

Things to critique

- Is the selection of tools provided by a given MCP server useful?
- Is it clear which tools were used and why? What arguments were passed in and what response did the LLM get back?
- Can you easily distinguish between a deterministic tool response and an LLM summary response?
- Did the LLM give an accurate account of what the tool gave back?
- What are some things that could go wrong with these kinds of Natural Language Interfaces to tools?
- How would the LLM perform an equivalent task without access to specialised tools?

Workshop resources

https://go.unimelb.edu.au/vh4h