Model Context Protocol (MCP)

Introduction

Copy page

Get started with the Model Context Protocol (MCP)

MCP is an open protocol that standardizes how applications provide context to LLMs. Think of MCP like a USB-C port for AI applications. Just as USB-C provides a standardized way to connect your devices to various peripherals and accessories, MCP provides a standardized way to connect AI models to different data sources and tools.

Why MCP?

MCP helps you build agents and complex workflows on top of LLMs. LLMs frequently need to integrate with data and tools, and MCP provides:

A growing list of pre-built integrations that your LLM can directly plug into

The flexibility to switch between LLM providers and vendors

Best practices for securing your data within your infrastructure

General architecture

At its core, MCP follows a client-server architecture where a host application can connect to multiple servers:

MCP Hosts: Programs like Claude Desktop, IDEs, or AI tools that want to access data through MCP

MCP Clients: Protocol clients that maintain 1:1 connections with servers

MCP Servers: Lightweight programs that each expose specific capabilities through the standardized Model Context Protocol

Local Data Sources: Your computer’s files, databases, and services that MCP servers can securely access

Remote Services: External systems available over the internet (e.g., through APIs) that MCP servers can connect to

Example MCP Servers

A list of example servers and implementations

This page showcases various Model Context Protocol (MCP) servers that demonstrate the protocol’s capabilities and versatility. These servers enable Large Language Models (LLMs) to securely access tools and data sources.

Reference implementations

These official reference servers demonstrate core MCP features and SDK usage:

Data and file systems

Filesystem - Secure file operations with configurable access controls

PostgreSQL - Read-only database access with schema inspection capabilities

SQLite - Database interaction and business intelligence features

Google Drive - File access and search capabilities for Google Drive

Development tools

Git - Tools to read, search, and manipulate Git repositories

GitHub - Repository management, file operations, and GitHub API integration

GitLab - GitLab API integration enabling project management

Sentry - Retrieving and analyzing issues from Sentry.io

Web and browser automation

Brave Search - Web and local search using Brave’s Search API

Fetch - Web content fetching and conversion optimized for LLM usage

Puppeteer - Browser automation and web scraping capabilities

Productivity and communication

Slack - Channel management and messaging capabilities

Google Maps - Location services, directions, and place details

Memory - Knowledge graph-based persistent memory system

AI and specialized tools

EverArt - AI image generation using various models

Sequential Thinking - Dynamic problem-solving through thought sequences

AWS KB Retrieval - Retrieval from AWS Knowledge Base using Bedrock Agent Runtime

Official integrations

These MCP servers are maintained by companies for their platforms:

Axiom - Query and analyze logs, traces, and event data using natural language

Browserbase - Automate browser interactions in the cloud

BrowserStack - Access BrowserStack's **Test Platform** to debug, write and fix tests, do accessibility testing and more.

Cloudflare - Deploy and manage resources on the Cloudflare developer platform

E2B - Execute code in secure cloud sandboxes

Neon - Interact with the Neon serverless Postgres platform

Obsidian Markdown Notes - Read and search through Markdown notes in Obsidian vaults

Prisma - Manage and interact with Prisma Postgres databases

Qdrant - Implement semantic memory using the Qdrant vector search engine

Raygun - Access crash reporting and monitoring data

Search1API - Unified API for search, crawling, and sitemaps

Stripe - Interact with the Stripe API

Tinybird - Interface with the Tinybird serverless ClickHouse platform

Weaviate - Enable Agentic RAG through your Weaviate collection(s)

Community highlights

A growing ecosystem of community-developed servers extends MCP's capabilities:

Docker - Manage containers, images, volumes, and networks

Kubernetes - Manage pods, deployments, and services

Linear - Project management and issue tracking

Snowflake - Interact with Snowflake databases

Spotify - Control Spotify playback and manage playlists

Todoist - Task management integration

Note: Community servers are untested and should be used at your own risk. They are not affiliated with or endorsed by Anthropic.

For a complete list of community servers, visit the **MCP Servers Repository**.

Getting started

Using reference servers

TypeScript-based servers can be used directly with `npx`:

Copy

```
npx -y @modelcontextprotocol/server-memory
```

Python-based servers can be used with `uvx` (recommended) or `pip`:

Copy

```
# Using uvx
uvx mcp-server-git

# Using pip
pip install mcp-server-git
python -m mcp_server_git
```

Configuring with Claude

To use an MCP server with Claude, add it to your configuration:

Copy

```
{
  "mcpServers": {
    "memory": {
      "command": "npx",
      "args": ["-y", "@modelcontextprotocol/server-memory"]
    },
    "filesystem": {
      "command": "npx",
      "args": ["-y", "@modelcontextprotocol/server-filesystem", "/
path/to/allowed/files"]
    },
    "github": {
      "command": "npx",
      "args": ["-y", "@modelcontextprotocol/server-github"],
      "env": {
        "GITHUB_PERSONAL_ACCESS_TOKEN": "<YOUR_TOKEN>"
      }
    }
  }
}
```

Additional resources

MCP Servers Repository - Complete collection of reference implementations and community servers

Awesome MCP Servers - Curated list of MCP servers

MCP CLI - Command-line inspector for testing MCP servers

MCP Get - Tool for installing and managing MCP servers

Pipedream MCP - MCP servers with built-in auth for 3,000+ APIs and 10,000+ tools

Supergateway - Run MCP stdio servers over SSE

Zapier MCP - MCP Server with over 7,000+ apps and 30,000+ actions

Visit our **GitHub Discussions** to engage with the MCP community.