Member-only story

A Visual Guide to Building a Streamable HTTP MCP Client

4 min read · Jul 17, 2025

Kartik Marwah Follow

Listen Share More

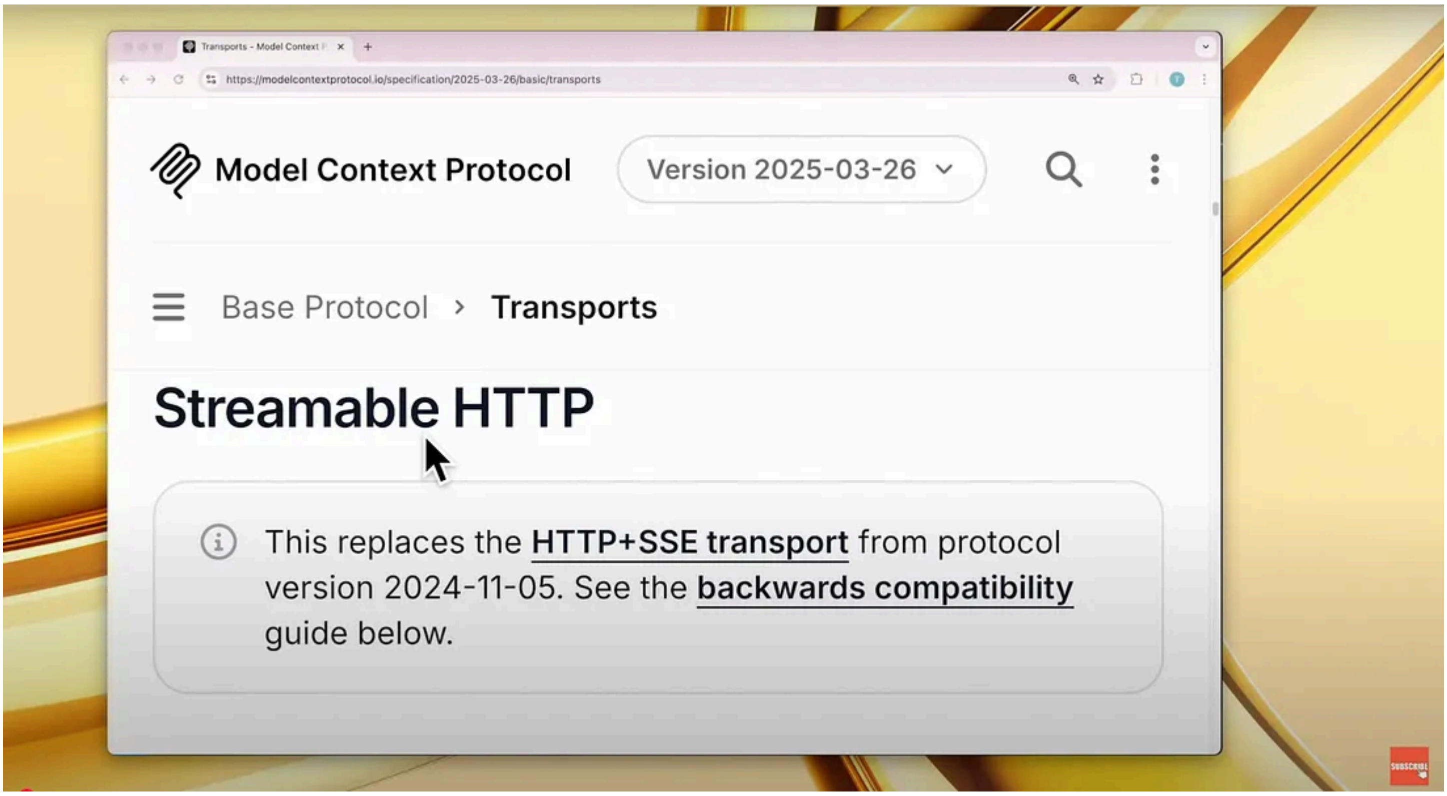
The Model Context Protocol (MCP) has introduced a powerful new **Streamable HTTP** transport, making it easier than ever to build and deploy agents that communicate over the internet.

This visual guide will walk you through creating a universal MCP client that connects to multiple tool servers using different transport methods. You can also watch a full video tutorial with code at The AI Language Channel here. (MCP Course #12 — Build Streamable HTTP + STDIO MCP Client with Gemini, Google ADK)

. . .

SSE vs. Streamable HTTP

The new Streamable HTTP transport is set to replace the older HTTP+SSE transport, offering a more robust and streamlined approach for remote tool communication.


Transports - Model Context P
https://modelcontextprotocol.io/specification/2025-03-26/basic/transports
Model Context Protocol
Version 2025-03-26
Base Protocol › Transports

Streamable HTTP

This replaces the HTTP+SSE transport from protocol version 2024-11-05. See the backwards compatibility guide below.

. . .

Part 1: The End Result — A Quick Demo

Our client will use three tool servers: two remote HTTP servers for math and one local STDIO server to run terminal commands. We'll give it a single, complex prompt and watch it orchestrate all the tools to get the job done.

First, we'll run the two HTTP servers and then start our universal client. The client immediately connects to all configured servers and lists the tools it has loaded from each one.

We’ll then give the agent a complex query to perform calculations using the HTTP servers and write the output to a file using the local terminal server.

The agent streams back its progress in real-time, showing each function call it makes.

Finally, the agent confirms it has completed the task and saved the results.

The run_command tool on our local terminal server successfully created a file with the formatted results.

Part 2: The Code — Understanding the Pieces

Our universal client is composed of three key python files and a config file.

1. The Entry Point (cmd.py)

This script acts as our command-line interface. It initializes the MCPClient and runs a continuous chat_loop to handle user input and display agent responses.

2. The Core Client Logic (client.py)

This file defines the MCPClient class, which is a wrapper that connects our UI to the Google ADK agent. It manages session creation, builds the agent, and sends tasks.

3. The Agent Brain (agent.py)

The AgentWrapper class here is responsible for building the LlmAgent. It loads all the toolsets from the servers defined in our config file and constructs the agent with a Gemini model.

4. Configuration is Key (theailanguage_config.json)

This JSON file is the heart of our client's configuration. It tells the agent which servers to connect to, specifying their type (http or stdio) and their connection details.

. . .

Part 3: Launching the Tool Servers

Now, let’s start our two streamable HTTP servers in separate terminals.

1. **Start Server 1** (listening on port 3000).

2. Start Server 2 (listening on port 3001).

. . .

Part 4: Running the Universal Client

With the servers running, we can now launch our client.

Once executed, the client starts up, connects to both HTTP servers and the local STDIO server, and confirms that all tools have been successfully loaded. You're now ready to interact with your multi-tool, multi-transport agent!

Mcp Client · Streamable Http · Adk · Model Context Protocol · Stdio Server

Follow

Published in The AI Language

23 followers · Last published Sep 17, 2025

Hands on coding samples to learn how to build Agents with Agent to Agent (A2A) Protocol, Model Context Protocol (MCP), Agent Development Kit. Use Free Gemini API Key to build. Deploy to Google Cloud Platform. Blog to our YT Channel https://youtube.com/@theailanguage

Follow

Written by Kartik Marwah

34 followers · 7 following

IIT Delhi and University of Illinois (UC) Alumnus with ~12 years of work experience in tech, consulting and building startups. Building "The AI Language"

No responses yet

Bgerby

What are your thoughts?

More from Kartik Marwah and The AI Language

In The AI Language by Kartik Marwah

A Visual Guide to MCP’s Streamable HTTP Transport

The world of AI agents and large language models (LLMs) is evolving rapidly, and so are the standards that govern how they communicate. One…

✦ Jun 25 👏 6 💬 1

In The AI Language by Kartik Marwah

Streamable HTTP MCP Server — Full Code Implementation

Build a Streamable HTTP MCP Server and test with Claude Desktop

✦ Jul 5 👏 5

In The AI Language by Kartik Marwah

A2A Protocol — Complete Overview

A deep dive into the lifecycle and core components of the Agent-to-Agent protocol and how it works along with MCP and ADK to build agentic...

✦ Jun 6 👏 52

In The AI Language by Kartik Marwah

I Coded a Real MCP + A2A Multi-Agent System From Scratch

A distributed, multi-agent system combining Google's A2A and Anthropic's MCP for complex problem-solving.

✦ Jul 30 7 4

See all from Kartik Marwah

See all from The AI Language

Recommended from Medium

In The AI Language by Kartik Marwah

A Visual Guide to MCP's Streamable HTTP Transport

The world of AI agents and large language models (LLMs) is evolving rapidly, and so are the standards that govern how they communicate. One...

✦ Jun 25 6 1

sai kiran varma

🚀 Building Production-Ready MCP Server with Streamable-HTTP Transport in 15 Minutes

The Developer Pain That Nobody Talks About

Jul 8 👏 4

Martin Hodges

Creating an MCP server to use in your agentic solution

This is the third article in a series that is exploring the use of AI to control a robot using human prompts in a solution I refer to as a…

Jul 6 👏 3

In Google Cloud - Community by Juan Guillermo Gómez Torres

How to expose FastAPI endpoints as tools with MCP and consume them from Gemini

Do you know how cool Large Language Models (LLMs) are? They can chat, write code, brainstorm ideas... but live in their digital world. What...

Jun 2 👏 5

In JavaScript in Plain English by Itsuki

Agent2Agent Protocol In Super Detail + Full Example(Server&Client) With OpenAI Agent As...

+ My interpretations, thoughts, and points to be careful about when implementing!

Jul 25 10

Heeki Park

Building an MCP server as an API developer

Anthropic released MCP at the end of November 2024. It took a few months for it to catch on, but my, oh my, it feels like the community is...

May 14 577 5

See more recommendations