MapQuest

MODEL CONTEXT PROTOCOL

MapQuest MCP Setup Guide

Connect MapQuest geocoding, routing, search, and traffic to Claude and any MCP-compatible AI agent in under five minutes.

MCP SERVER ENDPOINT

`https://www.mapquestapi.com/mcp`

SETUP PATHS INCLUDED

Cursor | Claude | Claude Code | ChatGPT | Codex

Version 1.2 · Updated July 7, 2026

BEFORE YOU BEGIN

Prerequisites & 5-minute quick start

The MapQuest MCP server exposes the Location Intelligence Suite (geocoding, routing, search, traffic, and truck-safe profiles) over the Model Context Protocol so your agent can call them directly. Every agent uses the same endpoint and the same OAuth flow; only the place you register it differs.

What you need for every agent

- A **MapQuest Developer account** — sign up at developer.mapquest.com/account/user/login/sign-up
- An **API key** — create one in your dashboard at developer.mapquest.com/account
- The **MCP server endpoint**: `https://www.mapquestapi.com/mcp` (identical for all agents)

The five steps, end to end

Register first so your key and free monthly transactions are active before your agent authorizes. Then pick your agent below and follow its path.

1. **Register and create your key**
   Create your MapQuest developer account, complete the account details, and generate the API key your agent will use.

2. **Choose your agent**
   Pick the agent you want to connect and follow its dedicated setup path in this guide.

3. **Add the MapQuest server**
   Paste the MapQuest endpoint or config in the place your agent expects it.

4. **Authorize MapQuest**
   Authorize from the connector, server entry, or tool settings for your agent via OAuth.

5. **Verify it in chat**
   Ask your agent which MCP tools are available.

Cursor

Cursor supports remote MCP servers through its built-in MCP settings. You add MapQuest by pasting a short entry into Cursor's mcp.json file.

Add the server

1. In Cursor, open **Settings** and go to **Tools & MCPs** (labeled **Tools & Integrations** in some versions, or **Tools** on the Free plan).
2. Under **MCP Tools**, click **New MCP Server**. Cursor opens your `mcp.json` file.

Add MapQuest inside the `mcpServers` object, then save. The key (`mapquest`) is the server name and the `url` is the endpoint:

JSON

```
{
  "mcpServers": {
    "mapquest": {
      "url": "https://www.mapquestapi.com/mcp"
    }
  }
}
```

Same entry for solo or team use; only the file location differs. Use `<project-root>/.cursor/mcp.json` to share it with a repo, or `~/.cursor/mcp.json` to make it available across all your projects.

Verify the connection

1. In **Tools & MCPs**, click **Connect** next to `mapquest`. Your browser opens the authorization page; approve it, then return to Cursor. If the server doesn't appear right away, restart Cursor and check again.
2. Confirm `mapquest` shows a connected status with its tools listed.
3. Open a new chat (Cmd + L or Ctrl + L) and ask: "What MCP tools do you have available?"

API keys. All tools require an active API key. On first tool use, the agent either auto-selects a key or prompts you to choose one by nickname.

Claude

The same setup works on claude.ai and the Claude desktop app (and on Cowork and mobile, where custom-connector install is in beta). MapQuest is a remote server, so you add it as a custom connector; you do not use the claude_desktop_config.json file (that is only for local servers and is not used by claude.ai or Cowork).

> **PLAN** Custom connectors are available on every plan: Free, Pro, Max, Team, and Enterprise. Free accounts are limited to one custom connector. On Team and Enterprise, an Owner or Primary Owner must add the connector for the organization first (in Organization settings > Connectors); members then connect to it individually.

Add the server

Add MapQuest as a custom connector:

1. Go to **Customize > Connectors** (claude.ai/customize/connectors).
2. Click the **+** button next to **Connectors**, then select **Add custom connector**.
3. Enter **Name** = `mapquest` and **URL** = `https://www.mapquestapi.com/mcp`. Leave **Advanced settings** (OAuth Client ID and Secret) empty unless MapQuest provides those values.
4. Click **Add**.
5. In the connector list, click **Connect** and complete the authorization in your browser, then return to Claude.

Verify the connection

1. Confirm `mapquest` shows a connected status under **Customize > Connectors**.
2. Open a new chat and enable it: click the **+** button in the lower left, hover **Connectors**, and toggle `mapquest` on.
3. Ask: "What MCP tools do you have available?"

API keys. All tools require an active API key. On first tool use, the agent either auto-selects a key or prompts you to choose one by nickname.

Claude Code

Claude Code is configured from the terminal, not the Connectors UI. You need Claude Code installed and signed in to your Claude account. Note: if you already added MapQuest as a connector in claude.ai and you sign in to Claude Code with that same account, it appears in Claude Code automatically (labeled as coming from claude.ai), and you can skip the steps below.

Add the server

Run from your terminal:

```bash
claude mcp add --transport http mapquest https://www.mapquestapi.com/mcp
```

This registers the server at **local** scope (just you, current project) by default. To change scope:

```bash
# Available across all your projects
claude mcp add --scope user --transport http mapquest https://www.mapquestapi.com/mcp

# Shared with teammates via a committed .mcp.json
claude mcp add --scope project --transport http mapquest https://www.mapquestapi.com/
mcp
```

Then authorize MapQuest. From your terminal, run `claude mcp login mapquest` to open your browser and sign in. (Inside a session you can instead run `/mcp`, which opens a panel listing your servers; select `mapquest` there and choose to authenticate, and your browser opens to finish.) If you configure servers by editing JSON instead of the command above, include `"type": "http"`; an entry with a URL but no type is read as a local server and fails to connect.

Verify the connection

1. Run `claude mcp list` (or `/mcp` in a session) and confirm `mapquest` shows a connected status.
2. Start a new session and ask: "What MCP tools do you have available?"

API keys. All tools require an active API key. On first tool use, the agent either auto-selects a key or prompts you to choose one by nickname.

ChatGPT

ChatGPT connects to MCP servers through custom apps (OpenAI renamed 'custom connectors' to 'apps' in December 2025). You add MapQuest as a new app in ChatGPT on the web.

> **PLAN** Custom MCP apps are set up in ChatGPT on the web. On Business, Enterprise, and Edu workspaces they are admin-controlled: an Owner or Admin enables Developer Mode and custom apps in Workspace settings, so if you don't see the option, ask your admin to turn it on.

Add the app

If custom apps are not already available in your account, first enable Developer Mode: open **Settings > Apps**, click **Advanced settings**, and toggle **Developer mode** on. Then create the app:

1. In **Settings > Apps**, create a new app. The dialog is titled **New App**.
2. **Name**: `mapquest`.
3. **Description** (optional): a short line such as “MapQuest geocoding, routing, search, and traffic tools.” ChatGPT uses this to help choose the right tool.
4. **Connection**: keep **Server URL** selected (the **Tunnel** tab is for local development) and enter `https://www.mapquestapi.com/mcp`.
5. **Authentication**: choose **OAuth**. Leave **Advanced OAuth settings** as they are unless MapQuest gives you a client ID and secret or specific scopes to enter.
6. Check **I understand and want to continue** to acknowledge the third-party risk notice, then click **Create**.

Verify the connection

1. On success, ChatGPT shows the tools the server advertises; complete the OAuth sign-in if prompted.
2. In a new chat, enable `mapquest` for the conversation from the composer's tools (Developer Mode) menu.
3. Ask: "What MCP tools do you have available?"

API keys. All tools require an active API key. On first tool use, the agent either auto-selects a key or prompts you to choose one by nickname.

Codex

Codex is configured through its config.toml file, which the Codex CLI and IDE extension share. You need Codex installed and signed in. MapQuest is a remote Streamable HTTP server, so you add it as a url entry and authenticate with OAuth.

Add the server

Codex runs in a terminal and is not preinstalled. Confirm it with `codex --version`; if that returns “command not found,” install it with `npm install -g @openai/codex` (requires Node.js), then run `codex login` to sign in. Set up MapQuest **before** launching Codex, or `/mcp` will look empty. Its config lives at `~/.codex/config.toml`; if that file does not exist yet, run `mkdir -p ~/.codex` first, then create it. To scope MapQuest to one trusted project instead, use `.codex/config.toml` there. In the IDE extension, open the file via **MCP settings > Open config.toml** from the gear menu. Add:

```toml
[mcp_servers.mapquest]
url = "https://www.mapquestapi.com/mcp"
```

Then authorize MapQuest over OAuth by running `codex mcp login mapquest` in your terminal; your browser opens to complete sign-in. Note: `codex mcp add` is for local stdio servers, so a remote HTTP server like MapQuest is added through config.toml as shown, not with that command.

Verify the connection

1. In the Codex TUI, run `/mcp` to confirm `mapquest` is listed and connected. Run `codex mcp --help` to see all MCP commands.
2. Start a session and ask: "What MCP tools do you have available?"

API keys. All tools require an active API key. On first tool use, the agent either auto-selects a key or prompts you to choose one by nickname.