COURSE SYLLABUS

Building Your First MCP Server and Client Course (Self-Paced)

Developers build a working MCP server and client from scratch using Python and FastMCP, learning both implementation skills and agent-native design principles for the Model Context Protocol.

> Group classes (Live Online) and onsite training are available for this course. For more information, email onsite@graduateschool.edu or visit: https://www.graduateschool.edu/courses/building-your-first-mcp-server-and-client-course-self-paced



support@graduateschool.edu •
(888) 744-4723

Course Outline

Module 1: Introduction to MCP and Agent-Facing Design

- Examine the integration problem MCP is designed to solve and how it standardizes connections between AI applications and external tools, resources, and workflows.
- Review MCP architecture, including the roles of the host, client, and server.
- Explore how context windows affect tool design and output design.
- Identify the three main server primitives: tools, resources, and prompts.

Module 2: Why Many MCP Servers Perform Poorly

- Analyze why MCP servers should not be designed as one-to-one REST wrappers and the cost of excessive tool discovery for agents.
- Evaluate why agent iteration is slower and more expensive than human trial and error.
- Assess how output format and schema size affect context efficiency.
- Apply principles of outcome-based tool design to improve server performance.

Module 3: Build an MCP Server with FastMCP

- Initialize a new FastMCP project, create a simple tool, and inspect the server with the MCP Inspector.
- Review the SpaceX API and identify likely user workflows for tool design.
- Build a naive version to observe anti-patterns, then refactor into a smaller, more effective toolset.
- Format outputs for LLM consumption using Markdown and improve tool descriptions, matching logic, and error handling.

Module 4: Build a Minimal MCP Client

- Connect to the server from a local client, discover tools via tools/list, and call tools programmatically.
- Examine the handshake and tool-call lifecycle between client and server.
- Observe how an LLM-driven host uses the same pattern to interact with MCP servers.

Module 5: Deployment, Security, and Ecosystem Overview

- Transition from local stdio development to remote deployment, including remote transport and hosting basics.
- Review security and authorization considerations for remote MCP servers.
- Survey the broader MCP ecosystem, including documentation, SDKs, and registries.