Beyond Function Calling: Understanding MCP

- **academy.kspl.tech | Koenig AI Academy**

Prerequisites check

- **You should already be able to explain a tool_use block, a tool input schema, and a tool_result.**
- **academy.kspl.tech | Koenig AI Academy**

What MCP adds

- MCP gives you three practical upgrades:
- Discovery: a host can ask a server what it offers instead of hard-coding every function in the
- Reuse: one MCP server can serve many hosts.
- Separation: the connector owner can maintain auth, data access, and domain logic outside the LL
- academy.kspl.tech | Koenig AI Academy

Host, client, server

- MCP uses three terms that are easy to blur:
- Host: the user-facing AI application, such as Claude Code or another agent UI.
- Client: the protocol component inside the host that speaks MCP.
- Server: the external process or service that exposes capabilities.
- academy.kspl.tech | Koenig AI Academy

Tools, resources, and prompts

- Tools are callable actions. Use tools for operations like search_invoices, create_support_ticke
- Resources are context objects the model can read. Use resources for project files, policy docum
- Prompts are reusable prompt templates exposed by the server. Use prompts when a connector knows
- model="claude-sonnet-4-6"
- academy.kspl.tech | Koenig AI Academy

Why this matters for connector design

- Bad MCP servers expose raw platform primitives: http_request, sql_query, run_command. Good MCP
- The connector designer's job is not to mirror every endpoint. It is to decide which operations
- academy.kspl.tech | Koenig AI Academy

Try it next

- **Deploy a gateway with RBAC, rate limits, and JSONL auditing**
- **academy.kspl.tech | Koenig AI Academy**