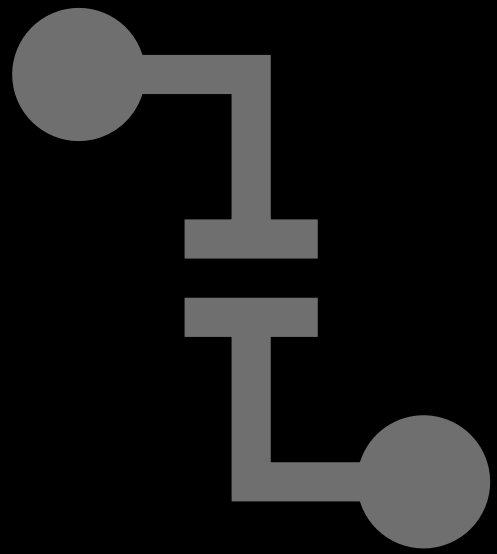
LONDON MCP

Lessons and Takeaways

ROADPLAN

- Discussion will focus on server-side MCP
- Discuss and explain MCP concepts
- Present MCP demo
- Discuss apptrack MCP setup
- Takeaways

WHAT IS THE MODEL CONTEXT PROTOCOL?



- MCP (Model Context Protocol) is an open-source standard for connecting AI applications to external systems

- Basically it is a standardized way to connect language models to functions and data

  - The result of that is to map language to action or intent

HOW DOES IT WORK

Mcp employs a messaging standard – JsonRPC – to standardize communication payloads

This makes it "transport-agnostic" which means these JsonRPC messages can be communicated over multiple different kinds of channels – even custom ones

At its core, mcp is a client-server messaging-format and communication-lifecycle standard

INITIAL CLIENT-SERVER "HANDSHAKE"

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "initialize",
  "params": {
    "protocolVersion": "2025-03-
26","capabilities": {
      "roots": {
        "listChanged": true
      },
      "sampling": {}
    },
    "clientInfo": {
      "name": "ExampleClient",
      "version": "1.0.0"
    }
  }
}
```

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "result": {
    "protocolVersion": "2025-03-26",
    "capabilities": {
      "logging": {},
      "prompts": {
        "listChanged": true
      },
      "resources": {
        "subscribe": true,
        "listChanged": true
      },
      "tools": {
        "listChanged": true
      }
    },
    "serverInfo": {
      "name": "ExampleServer",
      "version": "1.0.0"
    },
    "instructions": "Optional instructions for the
client"
  }
}
```

TOOLS: They are server-side functions made accessible to an mcp-client, typically automatically invoked by a language-model based on context

Often accept input-arguments whose format is defined through schemas

RESOURCES: They are server-side items, such as files or datasets, exposed to an mcp-client, typically explicitly requested by a user

Resource data can be exposed through raw text contents, or as a blob, dependiing on client-implementation

KEY MCP CONCEPTS

SCHEMAS AND ZOD

- SCHEMAS ARE AN IMPORTANT CONCEPT IN MCP, ESPECIALLY SINCE TOOL INPUT ARGUMENTS ARE BEING SYNTHESIZED BY AN LLM
- THE CONCEPT IS SIMILAR TO DATA-TRANSFER-OBJECT WITH REST API COMMUNICATION
- ZOD, A PEER DEPENDENCY OF THE MCP TS SDK, IS THE LIBRARY OF CHOICE TO DEFINE THESE SHCEMAS
- SCHEMA-VALIDATION BASED ON TOOL DEFINITIONS IS BUILT INTO THE MCP-SDK

DEMO

STATEFULLNESS AND SERIALIZABILITY

- Before moving to apptrack, it would be good to better establish these concepts in relation to the MCP
- Stateful MCP servers responses include the following two key headers:
  - 'Content-Type': 'text/event-stream'
  - Connection: 'keep-alive'
- This is essential to enable server-initiated events, such as notifications, but also features such as sampling and elicitation
- The mcp-session-id, handled out of the box with MCP SDK, is a helpful application tool for complex workflows, such as caching by key
- The major drawback of stateful MCP server implementation is a fundamental incompatibility with horizontal scaling, since the HTTP-transport object becomes non-serializable – due to circular object structures

APPTRACK MCP

APPTRACK MCP DISCUSSION

- Since apptrack has a complex workflow with many cross-validations, it made sense to group certain inputs together under one object
- We also had to becareful about overly restrictive input schemas that may throw a generically-formatted error at the MCP SDK level – before a tool callback is reached, but also be strict enough to define schemas that produce predictable inputs
- Leveraging zod checks for more complex validations, including cross validation scenarios, inside of a tool-callback proved a good approach for that
- Initially, I seperately defined types and tried to derive zod-schemas from them, however, this was not the correct approach as zod-schemas should be the source of truth
- Since we have a constants object that comes from backend which lists allowed values for some input fields, we had to generate zod constants with helper functions
- It is important to designate actionable errors that can be corrected and exposing them in our responses
- We can within the same application serve many different MCP servers on different urls

THANK YOU, QUESTIONS?