SPRING INTO MCP

Model Context Protocol for Java Developers

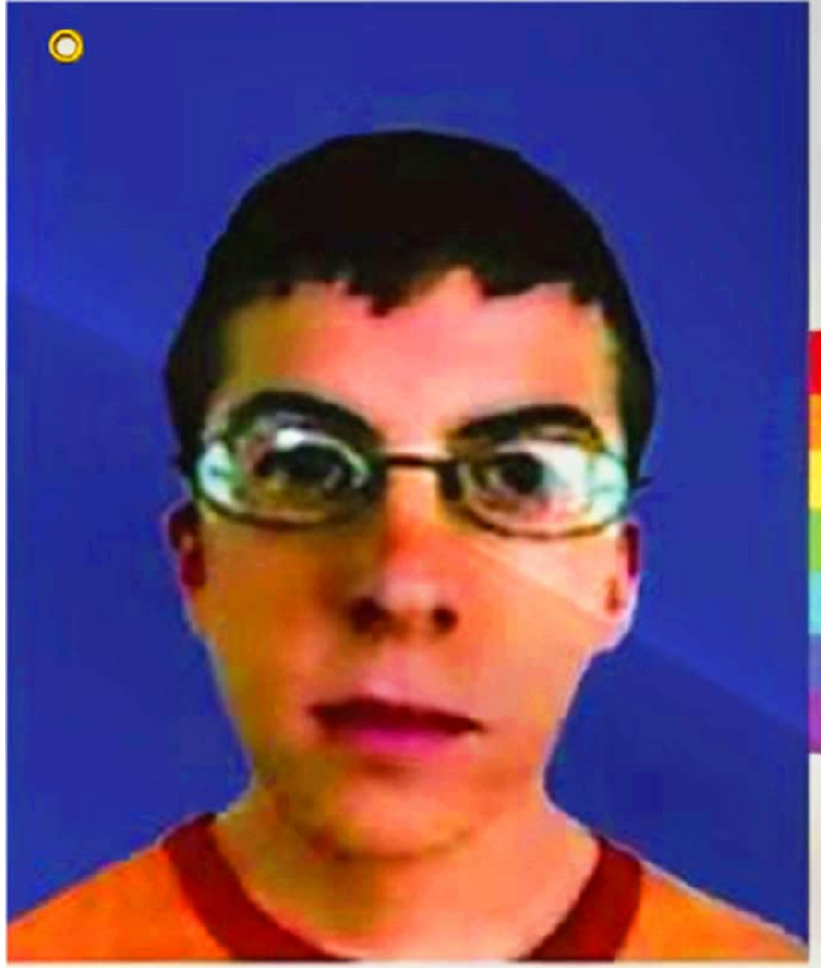
Dan Vega - Spring Developer Advocate @Broadcom

MCP

PROTOCOL

MC


Classic



HAWAII DRIVER LICENSE

NUMBER **01-47-87441**

DOB **06/03/1981** EXP **06/03/2008**

HT	WT	HAIR	EYES	SEX	CTY
5-10	150	BRO	BRO	M	0

ISSUE DATE	CLASS	RESTR	ENDORSE
06/18/1998	3		

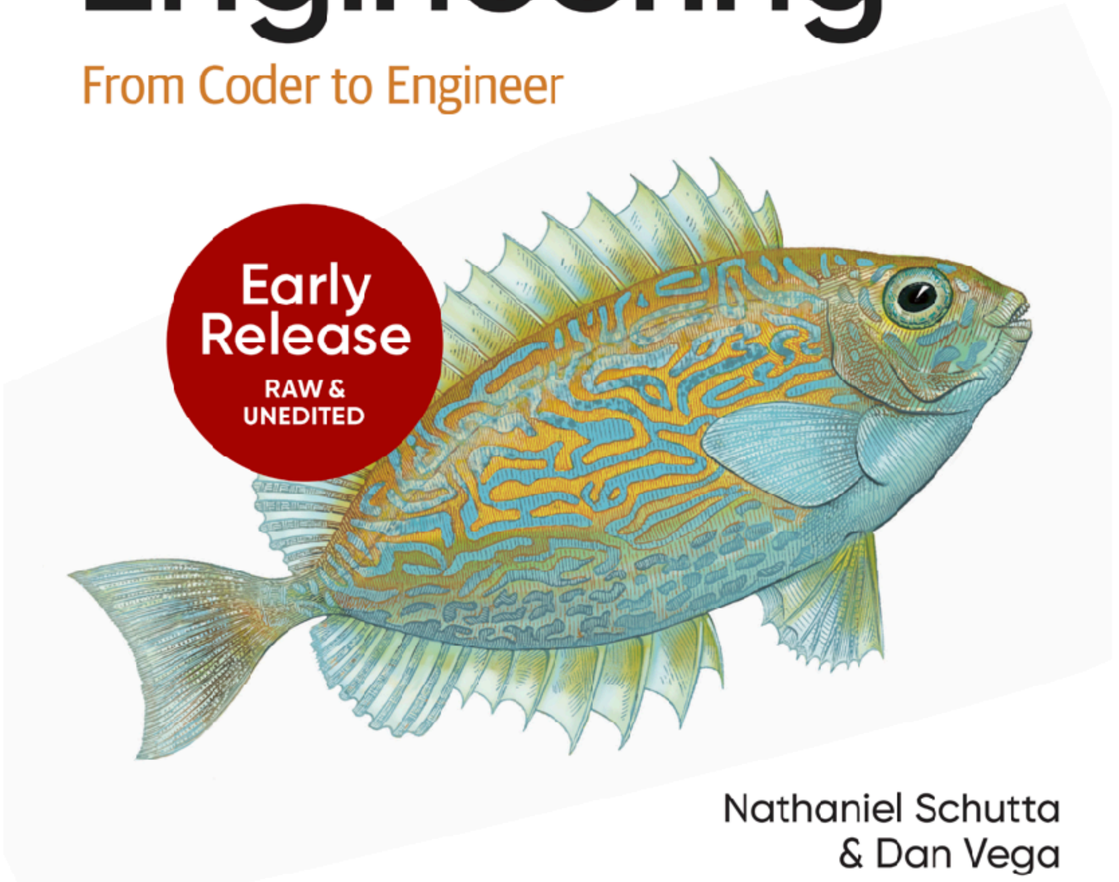
McLOVIN
892 MOMONA ST
HONOLULU, HI 96820

MODEL CONTEXT PROTOCOL

ABOUT ME

Learn more at danvega.dev

- 👨‍👩‍👧‍👦 Husband & Father
- 🏠 Cleveland
- ☕ Java Champion
- 👨‍💻 Software Development 23 Years
- 🍃 Spring Developer Advocate
- 📖 Author (Soon to be)



AGENDA

- LLM Limitations
- Model Context Protocol
- Build MCP Servers in Java / Spring
- DEMO
- Personal Real World Use Case

LLM LIMITATIONS

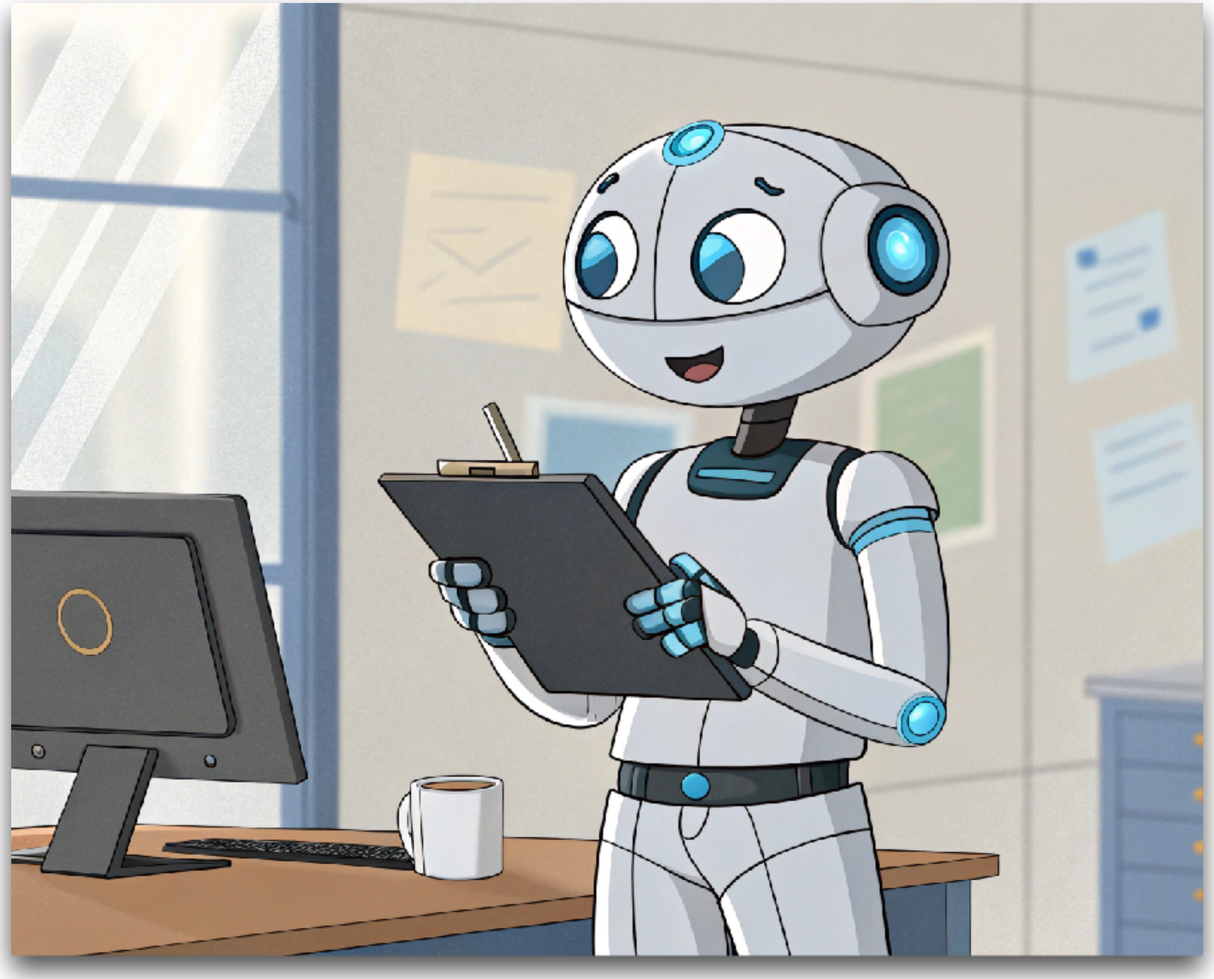
WHY DO WE CARE?

LLMs are like super-smart interns

Risk of Wrong Answers

Risk of Lost Trust

Run-away Costs

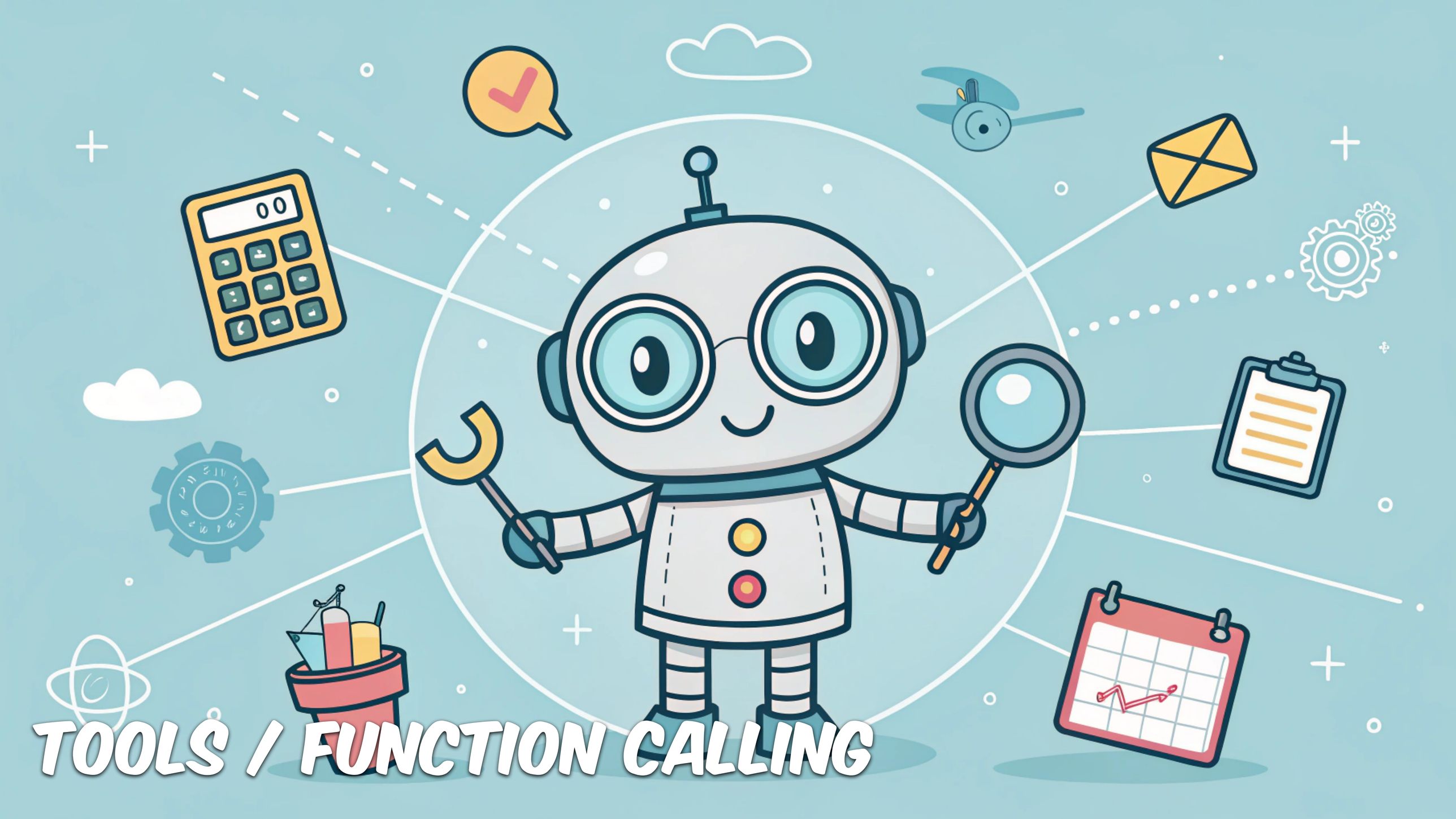


Limitation	Quick Note — “what this looks like”
🧠✨ **Hallucinate**	Invents facts or API names with total confidence
🕒 **Stale Data**	Knowledge cutoff means “today’s stock price?” → ¯_(ツ)_/¯
⚖️ **Bias & Safety**	Outputs stereotypes, toxic language, or policy violations
🏗️ **Domain Gaps**	Uses generic wording where niche jargon is required
📏 **Context Window**	Long threads get truncated; model “forgets” earlier details
🎲 **Non-Determinism**	Same prompt, different answer → flaky tests, review churn
🔒 **Privacy Leak**	Proprietary/PII text could leave your trusted boundary
💰⏱️ **Cost & Latency**	High-token chains drain budget and slow UX
🧩 **Weak Reasoning**	Multi-step calculations or logical deductions fail
🔍 **Low Explainability**	Hard to audit: “Why did you choose that answer?”

LLM LIMITATIONS

Here's our Swiss-army lineup for taming any limitation

#	Lever	Purpose
1	🛡️ **Prompt Guarding**	Encode rules that constrain the model's behavior (tone, honesty, refusal policy).
2	📄 **Prompt Stuffing / RAG**	Inject fresh, task-specific context so the model quotes facts instead of guessing.
3	🔧 **Tools / Function Calling**	Let the model invoke code or APIs for real-time data, calculations, or business logic.
4	🌐 **MCP (Resources + Tools)**	Package those tools as reusable, versioned endpoints every client can share.


TOOLS / FUNCTION CALLING

TOOL CALLING / FUNCTION CALLING

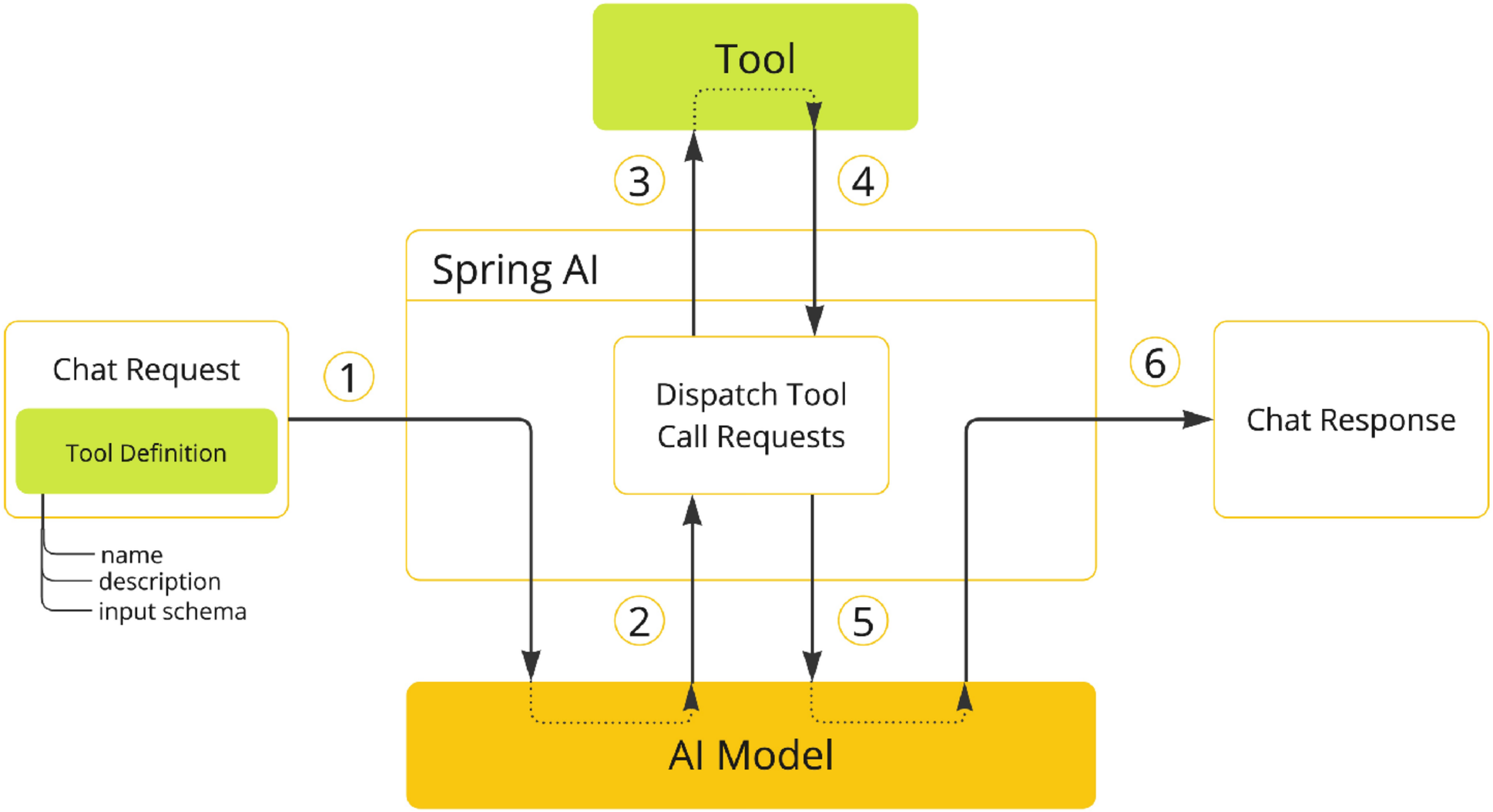
Information Retrieval

Tools in this category can be used to retrieve information from external sources, such as a database, a web service, a file system, or a web search engine. The goal is to augment the knowledge of the model, allowing it to answer questions that it would not be able to answer otherwise. As such, they can be used in Retrieval Augmented Generation (RAG) scenarios. For example, a tool can be used to retrieve the current weather for a given location, to retrieve the latest news articles, or to query a database for a specific record.

TOOL CALLING / FUNCTION CALLING

Taking Action

Tools in this category can be used to take action in a software system, such as sending an email, creating a new record in a database, submitting a form, or triggering a workflow. The goal is to automate tasks that would otherwise require human intervention or explicit programming. For example, a tool can be used to book a flight for a customer interacting with a chatbot, to fill out a form on a web page, or to implement a Java class based on an automated test (TDD) in a code generation scenario.


Tool
3
4
Spring AI
Chat Request
Tool Definition
name
description
input schema
1
Dispatch Tool
Call Requests
6
Chat Response
2
5
AI Model

```java
public class DatTimeTools {

    @Tool(description = "Get the current date and time in the user's timezone")
    String getCurrentDateTime() {
        return LocalDateTime.now().atZone(LocaleContextHolder.getTimeZone().toZoneId()).toString();
    }

}
```

MODEL CONTEXT PROTOCOL

spring

WHAT IS MODEL CONTEXT PROTOCOL (MCP)?

MCP is a protocol that defines how to talk to AI models in a consistent, structured way.

MOTIVATION

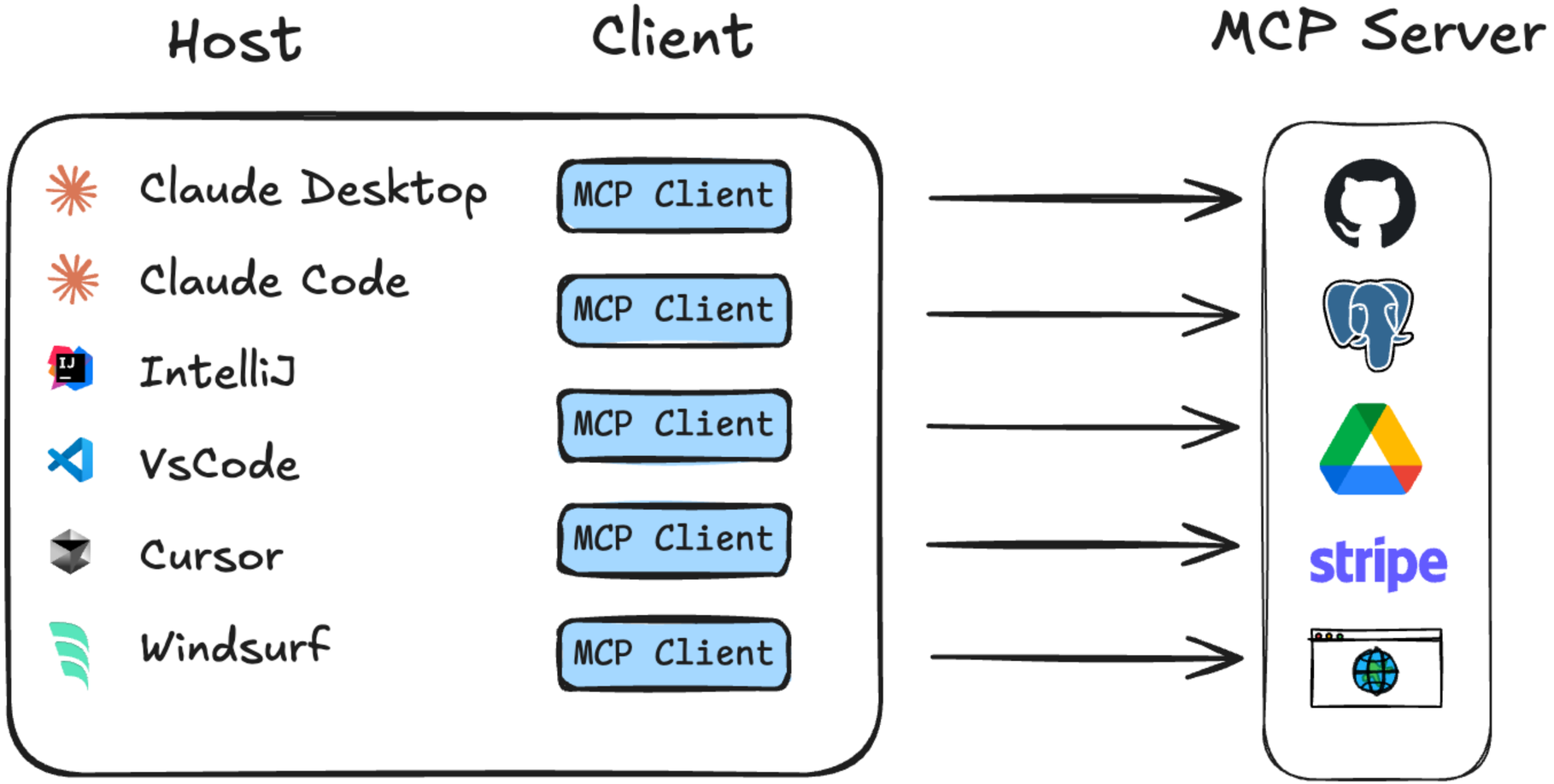
Models are only as good as the **context** you provide them

BENEFITS OF MCP

- **Modularity:** 🧠 This lets the AI app stay light while still being deeply aware of your environment.
- **Reusability:** 💡 Once you write an MCP server (say, one that surfaces project files), it can serve any app that supports the MCP protocol.
- **Language Agnostic:** 👨🏻‍💻 Works across programming languages and environments without language-specific dependencies.
- **Fine-Grained Control:** 🔐 It gives you programmable control over what your AI sees and can do
- **Privacy and Security:** 🛡️ You stay in control of what your AI knows.

COMPONENTS OF MCP

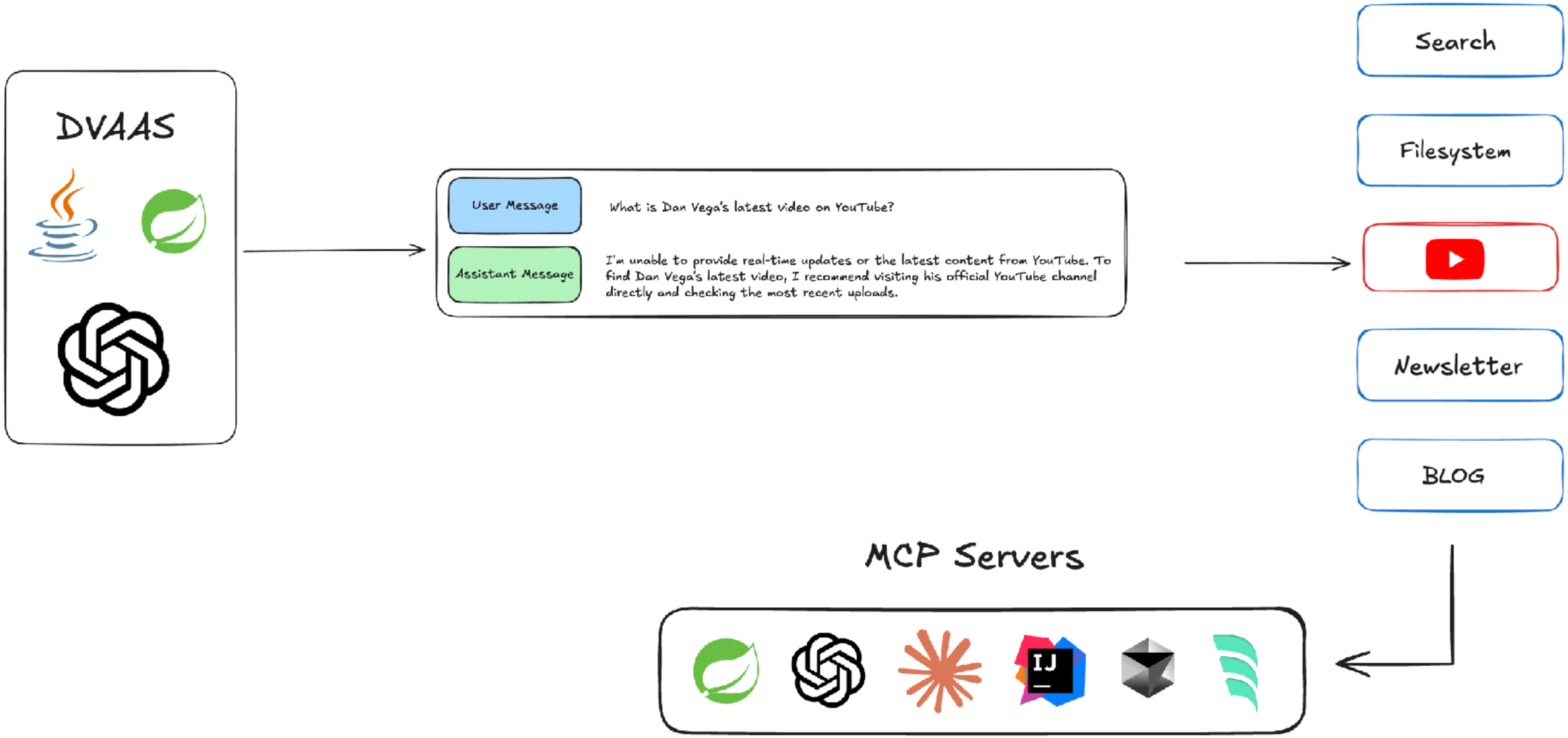
Participants



HOW IS AN MCP SERVER DIFFERENT THAN AN API?

Provider Agnostic

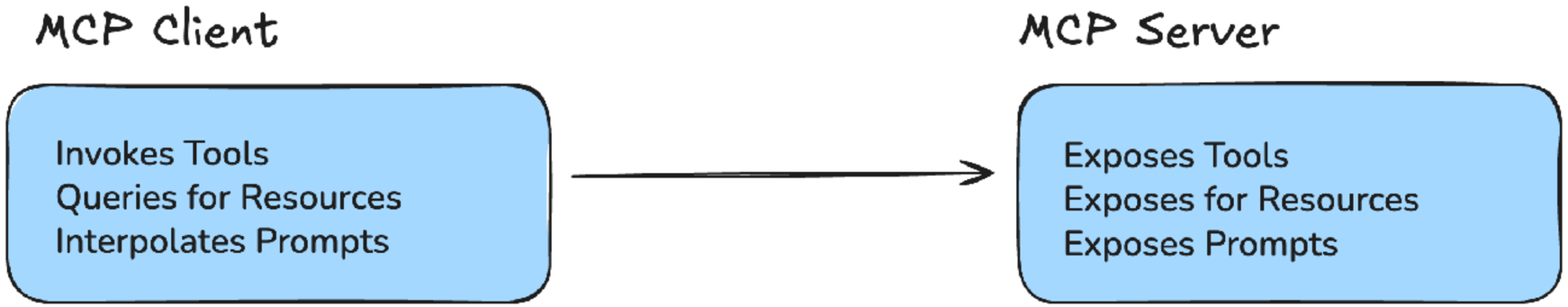
HOW IS THIS DIFFERENT FROM TOOL USE?



PRIMITIVES

MCP PRIMITIVES

MCP primitives are the most important concept within MCP. They define what clients and servers can offer each other. These primitives specify the types of contextual information that can be shared with AI applications and the range of actions that can be performed.



MCP PRIMITIVES: TOOLS

What: Executable functions that AI applications can invoke to perform actions (e.g., file operations, API calls, database queries)

When: AI needs to take action beyond just generating text - when it needs to DO something in the real world

Examples:

- read_file() - Read contents of a specific file
- write_file() - Create or modify files
- execute_sql() - Run database queries
- send_email() - Send messages via email API
- git_commit() - Commit changes to repository
- slack_post() - Send messages to Slack channels
- web_search() - Search the internet
- calculate() - Perform mathematical operations

MCP PRIMITIVES: RESOURCES

What: Data sources that provide contextual information to AI applications (e.g., file contents, database records, API responses)

When: AI needs to understand or reference existing information before responding or taking action

Examples:

- file://project/README.md - Documentation and project files
- database://users/profile/123 - User records and data
- git://repo/commit/history - Version control information
- slack://channel/messages - Chat history and conversations
- calendar://events/today - Schedule and meeting data
- email://inbox/recent - Email content and metadata

MCP PRIMITIVES: PROMPTS

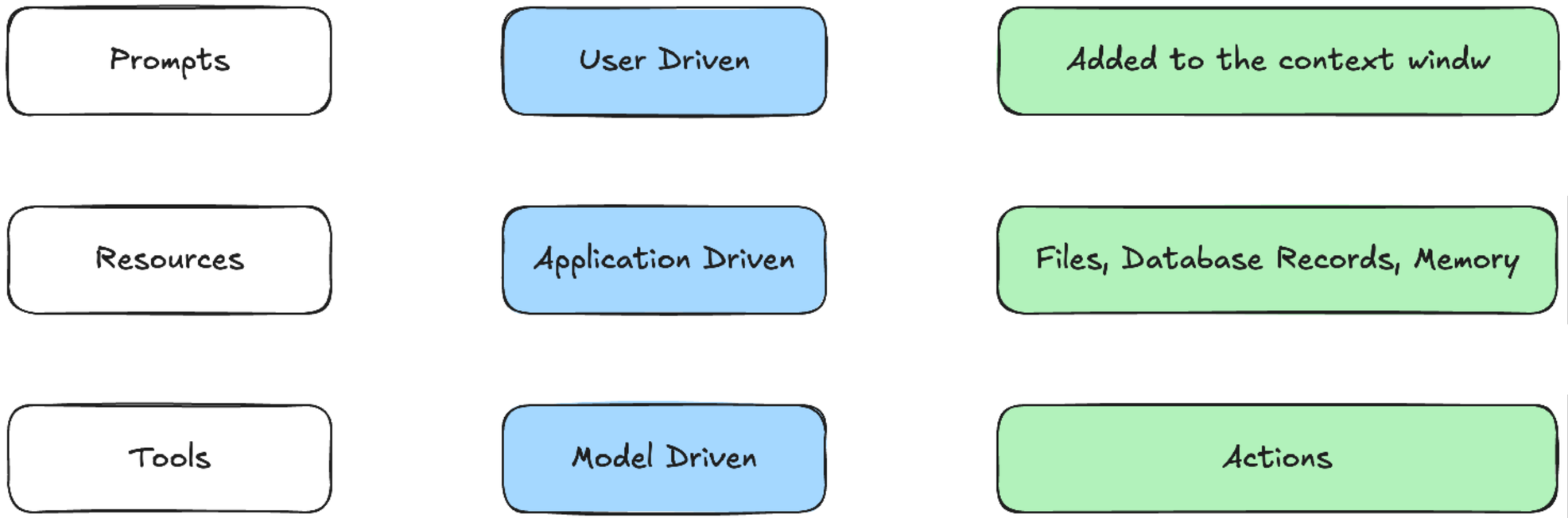
What: Reusable templates that help structure interactions with language models (e.g., system prompts, few-shot examples)

When: You need consistent, well-crafted prompts across different conversations or want to standardize AI behavior patterns

Examples:

- code_reviewer - Template for reviewing pull requests with specific criteria
- meeting_summarizer - Structured format for extracting action items from transcripts
- technical_writer - Guidelines for creating documentation in company style
- bug_triager - Template for categorizing and prioritizing issues
- customer_support - Consistent tone and approach for user interactions
- data_analyst - Framework for interpreting charts and metrics
- project_planner - Structure for breaking down tasks and timelines
- content_moderator - Guidelines for evaluating user-generated content

MCP PRIMITIVES: INTERACTION MODEL



TRANSPORTS

Transports in the Model Context Protocol (MCP) provide the foundation for communication between clients and servers. A transport handles the underlying mechanics of how messages are sent and received.

TRANSPORTS

Standard Input/Output (stdio)

- Use stdio when:
  - Building command-line tools
  - Implementing local integrations
  - Needing simple process communication
  - Working with shell scripts

Server-Sent Events (SSE)

- Use SSE when:
  - Only server-to-client streaming is needed
  - Working with restricted networks
  - Implementing simple updates

AUTHORIZATION

AUTHORIZATION ALLOWS:

Allows private context to be shared from trusted data sources

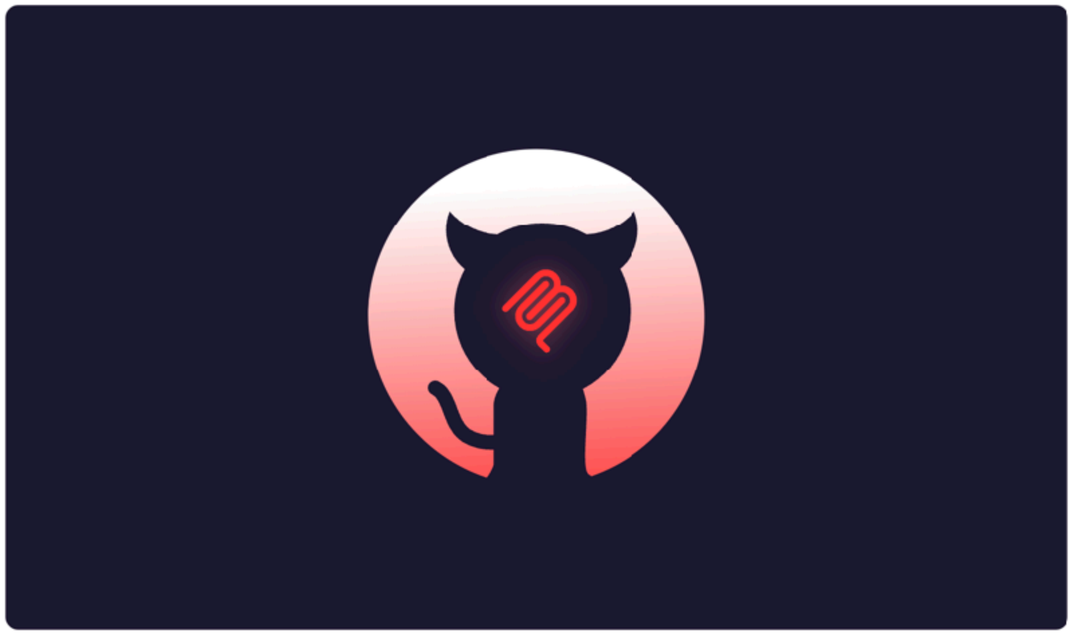
Enable MCP server authors to bind the capabilities of a server to an account.

Securely connect to third-party integrations.

2025-05-26

GitHub MCP Exploited: Accessing private repositories via MCP

We showcase a critical vulnerability with the official GitHub MCP server, allowing attackers to access private repository data. The vulnerability is among the first discovered by Invariant's security analyzer for detecting toxic agent flows.

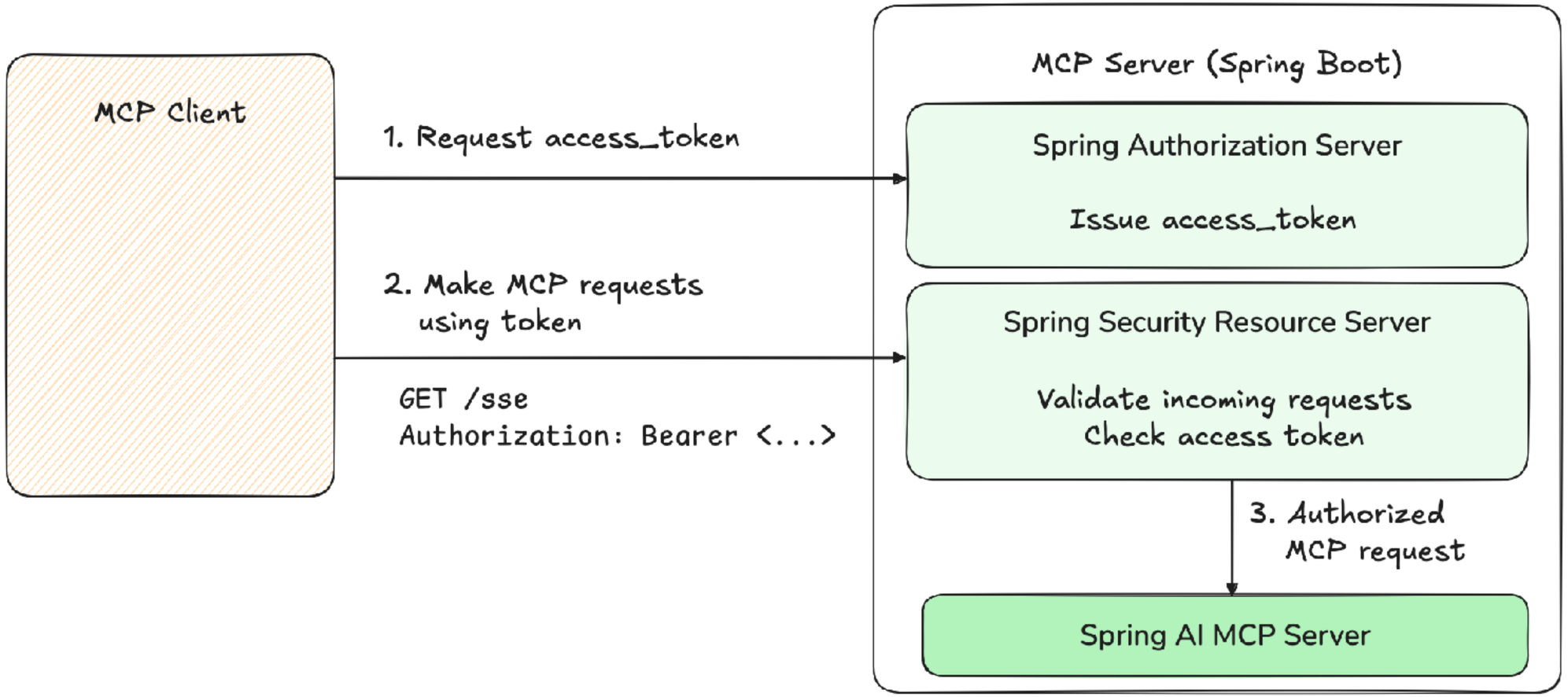


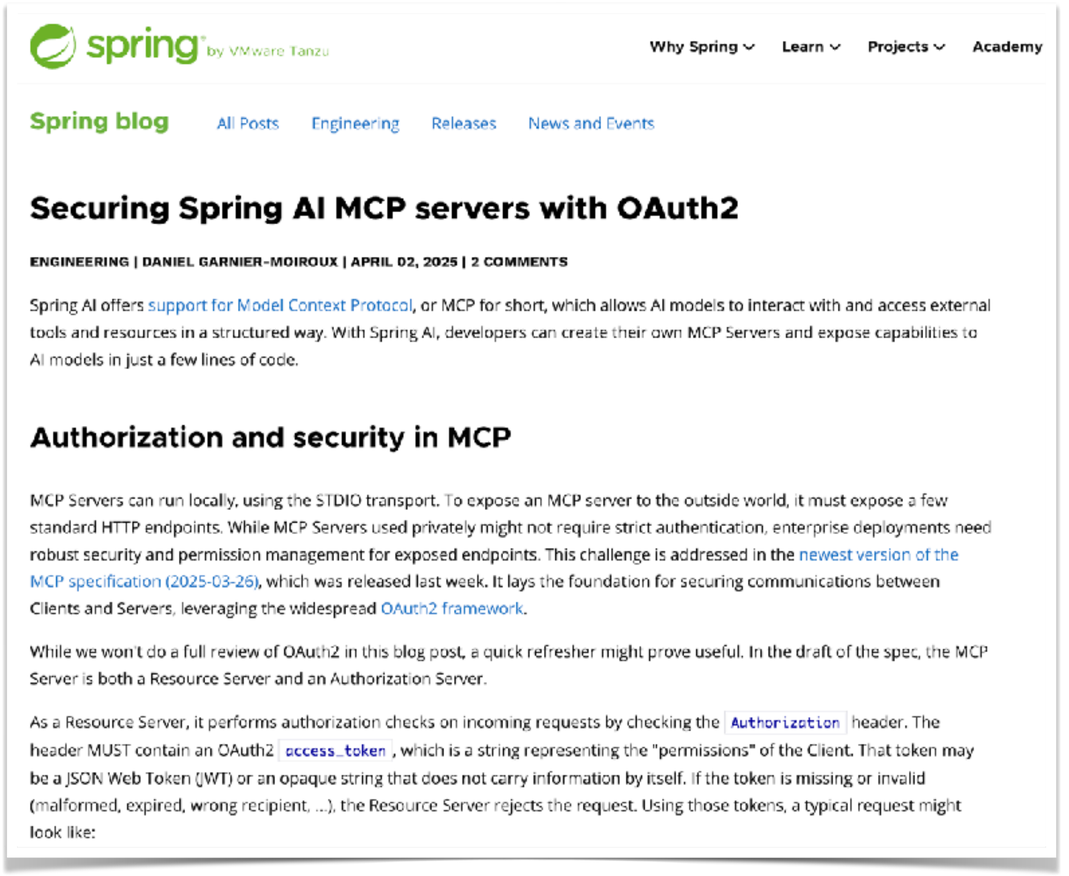
Invariant has discovered a critical vulnerability affecting the widely-used GitHub MCP integration (14k stars on GitHub). The vulnerability allows an attacker to hijack a user's agent via a malicious GitHub Issue, and coerce it into leaking data from private repositories.

SECURING MCP SERVERS

- **Security Challenge:** While local MCP servers (STDIO transport) may not need authentication, enterprise HTTP deployments require robust security and permission management
- **OAuth2 Integration:** New MCP spec (2025-03-26) leverages OAuth2 framework - MCP server acts as both Resource Server (validates tokens) and Authorization Server (issues tokens)
- **Implementation Requirements:**
  - Add Spring Security & Spring Authorization Server Dependencies
  - Configure OAuth2 client credentials in application.properties
  - Create SecurityFilterChain to handle authentication and token validation

SECURING MCP SERVERS





spring by VMware Tanzu

Why Spring Learn Projects Academy

Spring blog All Posts Engineering Releases News and Events

Securing Spring AI MCP servers with OAuth2

ENGINEERING | DANIEL GARNIER-MOIROUX | APRIL 02, 2025 | 2 COMMENTS

Spring AI offers support for Model Context Protocol, or MCP for short, which allows AI models to interact with and access external tools and resources in a structured way. With Spring AI, developers can create their own MCP Servers and expose capabilities to AI models in just a few lines of code.

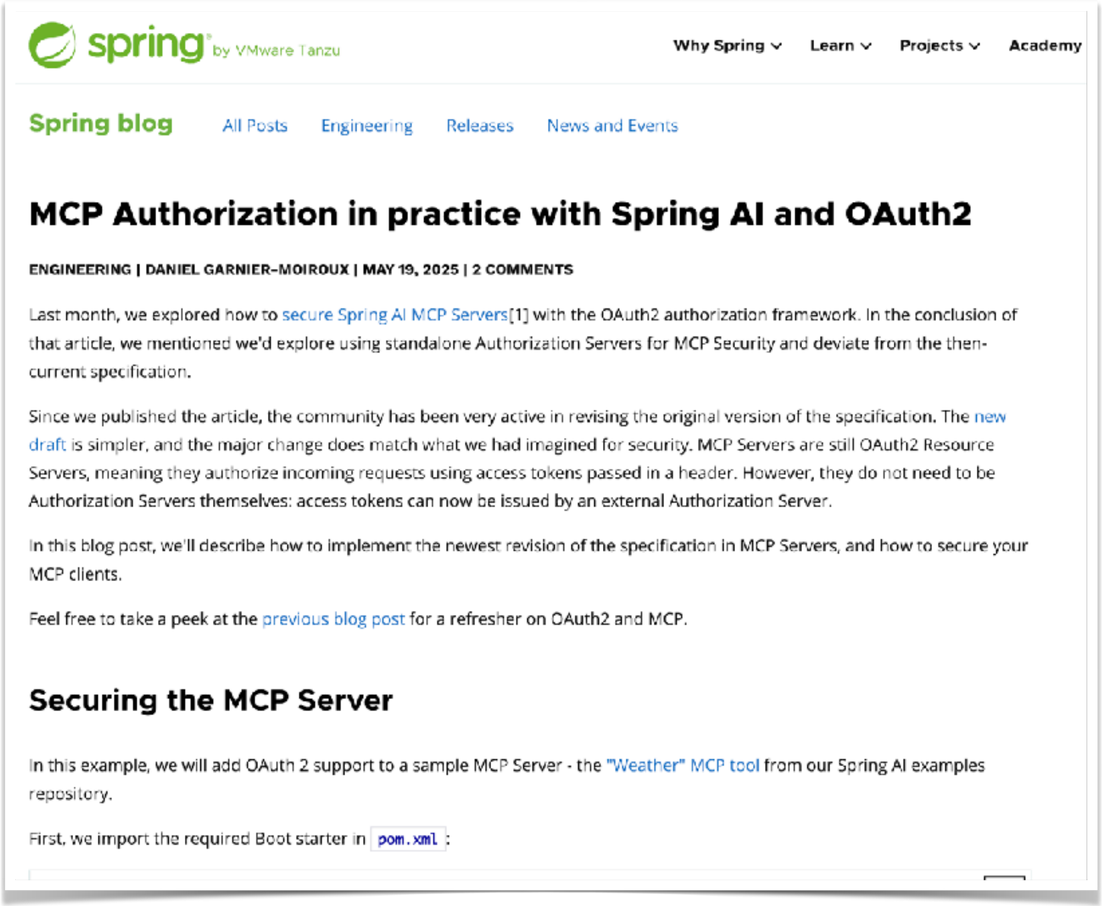
Authorization and security in MCP

MCP Servers can run locally, using the STDIO transport. To expose an MCP server to the outside world, it must expose a few standard HTTP endpoints. While MCP Servers used privately might not require strict authentication, enterprise deployments need robust security and permission management for exposed endpoints. This challenge is addressed in the newest version of the MCP specification (2025-03-26), which was released last week. It lays the foundation for securing communications between Clients and Servers, leveraging the widespread OAuth2 framework.

While we won't do a full review of OAuth2 in this blog post, a quick refresher might prove useful. In the draft of the spec, the MCP Server is both a Resource Server and an Authorization Server.

As a Resource Server, it performs authorization checks on incoming requests by checking the `Authorization` header. The header MUST contain an OAuth2 `access_token`, which is a string representing the "permissions" of the Client. That token may be a JSON Web Token (JWT) or an opaque string that does not carry information by itself. If the token is missing or invalid (malformed, expired, wrong recipient, ...), the Resource Server rejects the request. Using those tokens, a typical request might look like:

https://spring.io/blog/2025/04/02/mcp-server-oauth2



spring by VMware Tanzu

Why Spring Learn Projects Academy

Spring blog All Posts Engineering Releases News and Events

MCP Authorization in practice with Spring AI and OAuth2

ENGINEERING | DANIEL GARNIER-MOIROUX | MAY 19, 2025 | 2 COMMENTS

Last month, we explored how to secure Spring AI MCP Servers[1] with the OAuth2 authorization framework. In the conclusion of that article, we mentioned we'd explore using standalone Authorization Servers for MCP Security and deviate from the then-current specification.

Since we published the article, the community has been very active in revising the original version of the specification. The new draft is simpler, and the major change does match what we had imagined for security. MCP Servers are still OAuth2 Resource Servers, meaning they authorize incoming requests using access tokens passed in a header. However, they do not need to be Authorization Servers themselves: access tokens can now be issued by an external Authorization Server.

In this blog post, we'll describe how to implement the newest revision of the specification in MCP Servers, and how to secure your MCP clients.

Feel free to take a peek at the previous blog post for a refresher on OAuth2 and MCP.

Securing the MCP Server

In this example, we will add OAuth 2 support to a sample MCP Server - the "Weather" MCP tool from our Spring AI examples repository.

First, we import the required Boot starter in `pom.xml`:

https://spring.io/blog/2025/05/19/spring-ai-mcp-client-oauth2

BUILDING MCP SERVERS IN JAVA / SPRING

spring®

SDKs

Copy page

Official SDKs for building with the Model Context Protocol

Build MCP servers and clients using our official SDKs. Choose the SDK that matches your technology stack - all SDKs provide the same core functionality and full protocol support.

Available SDKs

- TypeScript
- Python
- Go
- Kotlin
- Swift
- Java
- C#
- Ruby
- Rust

spring initializr

Project

- ○ Gradle - Groovy
- ○ Gradle - Kotlin
- ● Maven

Language

- ● Java
- ○ Kotlin
- ○ Groovy

Spring Boot

- ○ 4.0.0 (SNAPSHOT)
- ○ 4.0.0 (M1)
- ○ 3.5.5 (SNAPSHOT)
- ● 3.5.4
- ○ 3.4.9 (SNAPSHOT)
- ○ 3.4.8

Project Metadata

Group	dev.danvega
Artifact	kcdc-mcp
Name	kcdc-mcp
Description	Demo project for Spring Boot
Package name	dev.danvega.kcdc
Packaging	● Jar ○ War
Java	● 24 ○ 21 ○ 17

Dependencies ADD DEPENDENCIES... ⌘ + B

Spring Web WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Model Context Protocol Server AI

Spring AI support for Model Context Protocol (MCP) servers.

GENERATE ⌘ + ↵ EXPLORE CTRL + SPACE ...

BUILDING AN MCP SERVER IN JAVA / SPRING

Spring Programming Model

With some additional MCP APIs

- Tools
- Prompts
- Resources

```java
@Component
public class SessionTools {

    private static final Logger log = LoggerFactory.getLogger(SessionTools.class);
    private Conference conference;
    private final ObjectMapper objectMapper;

    public SessionTools(ObjectMapper objectMapper) {
        this.objectMapper = objectMapper;
    }

    @Tool(name = "kcdc-sessions", description = "Returns all sessions for KCDC 2025")
    public List<Session> findAllSessions() {
        return conference.sessions();
    }

    @PostConstruct
    public void init() {
        log.info("Loading Sessions from JSON file 'sessions.json'");
        try (InputStream inputStream = TypeReference.class.getResourceAsStream("/data/sessions.json")) {
            this.conference = objectMapper.readValue(inputStream, Conference.class);
        } catch (IOException e) {
            throw new RuntimeException("Failed to read JSON data", e);
        }
    }

}
```

```java
@SpringBootTest
class SessionToolsTest {

    @Autowired
    SessionTools sessionTools;

    @Test
    void testCountSessionsByDate() {
        Map<String, Long> sessionsByDate = sessionTools.countSessionsByDate();

        assertThat(sessionsByDate).isNotEmpty();
        assertThat(sessionsByDate.keySet()).allMatch(date -> date != null && !date.isEmpty());
        assertThat(sessionsByDate.values()).allMatch(count -> count > 0);

        // Verify we have sessions for KCDC 2025 dates
        assertThat(sessionsByDate.keySet()).contains("2025-08-13", "2025-08-14", "2025-08-15");
    }

}
```

TESTING YOUR MCP SERVERS

- Create an executable JAR
- Test using an MCP Client
  - Spring MCP Client
  - Claude Desktop
  - Cursor / Windsurf / Junie
  - Any MCP Client of your choice

MCP Inspector v0.14.2

Transport Type

STDIO

Command

/Users/vega/.sdkman/candidates/

Arguments

-jar /Users/vega/Downloads/spri

Environment Variables

Server Entry | Servers File

Configuration

Restart | Disconnect

Connected

Logging Level

debug

System

Resources | Prompts | Tools | Ping | Sampling | Roots | Auth

Tools

List Tools

Clear

spring-io-sessions

Returns all sessions for Spring I/O 2025 Conference

spring-io-sessions

Returns all sessions for Spring I/O 2025 Conference

Run Tool

Tool Result: Success


```
[
  0: {
    day: "2025-05-22"
    time: "08:00"
    title: "Registration"
    type: "logistics"
    speakers: []
    room: "General"
  }
  1: {
    day: "2025-05-22"
    time: "09:00"
    title: "Welcome"
    type: "opening"
    speakers: [
```


History

6. ping ▶

5. initialize ▶

4. tools/call ▶

3. tools/list ▶

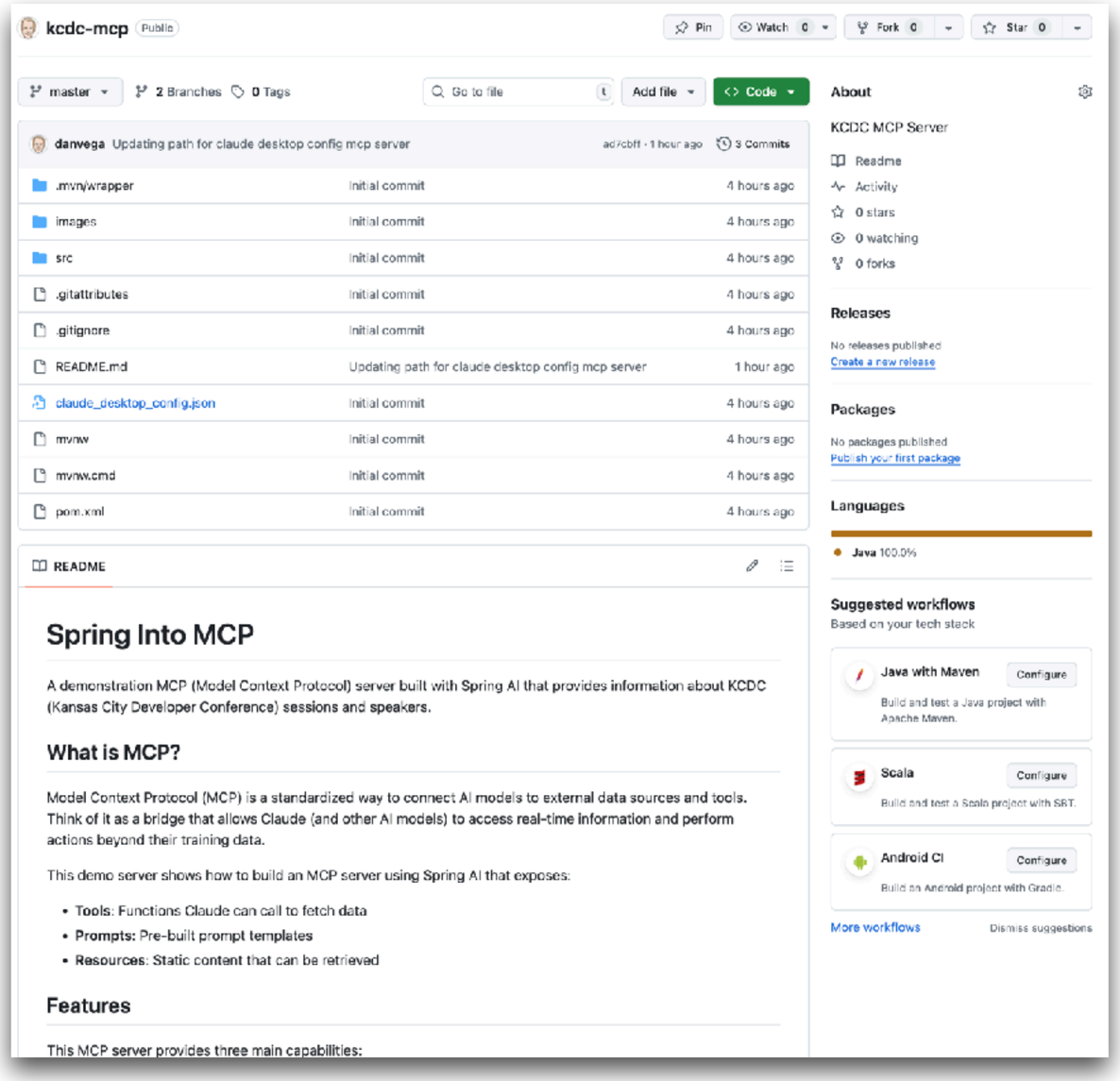
2. resources/list ▶

Server Notifications

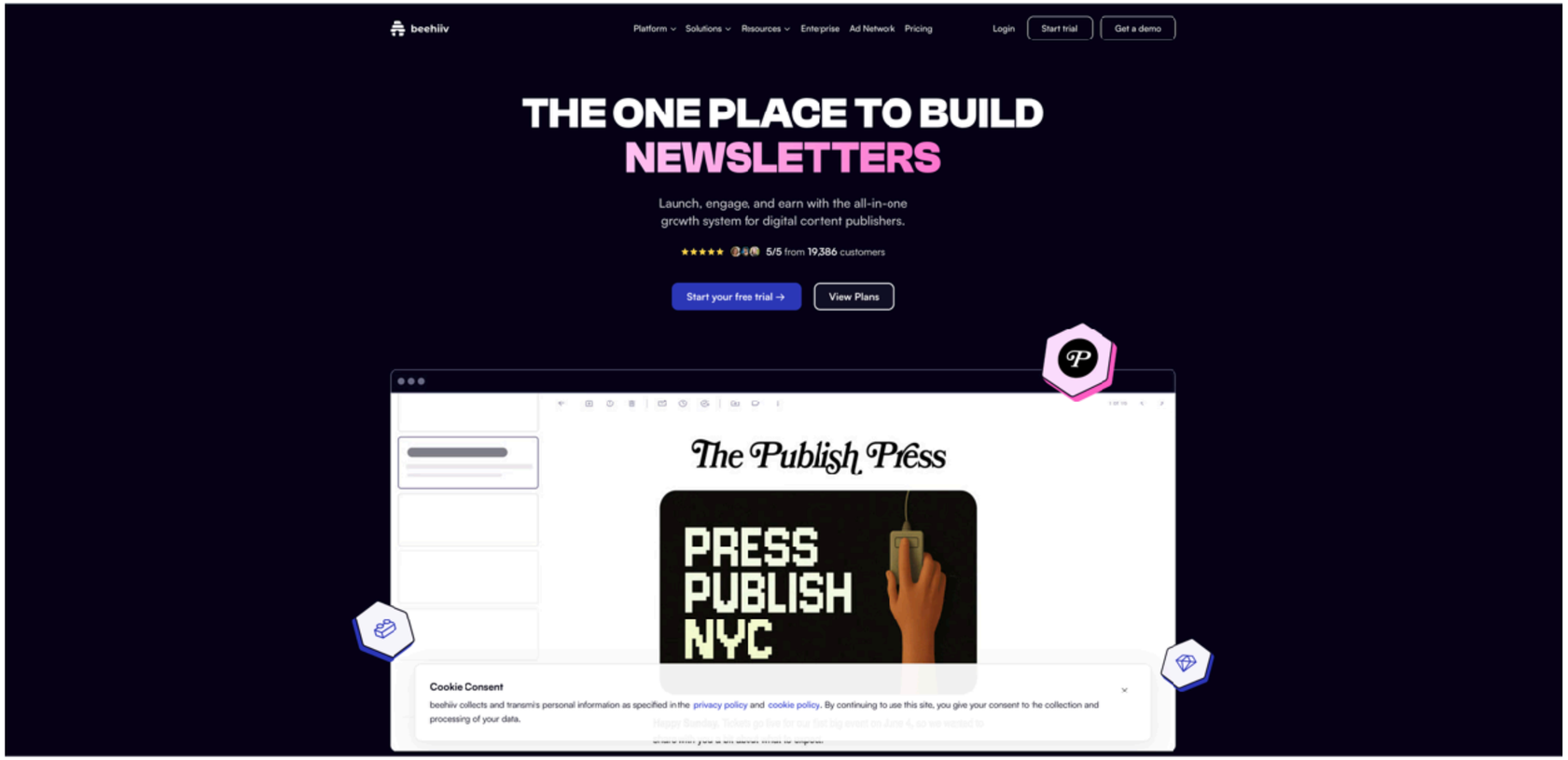
No notifications yet

DEMO TIME

https://github.com/danvega/kcdc-mcp



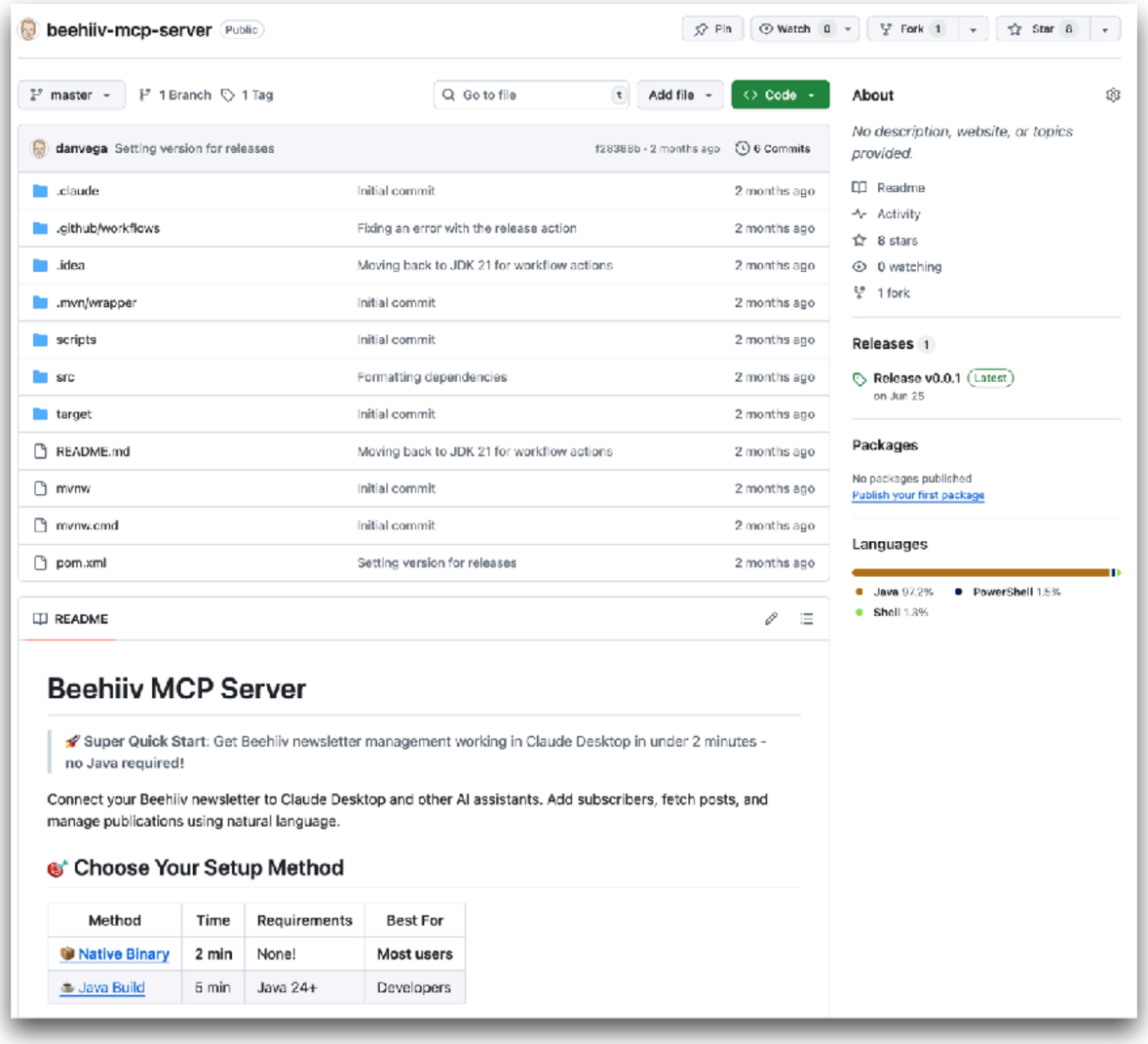
BEEHIIV



https://github.com/danvega/beehiiv-mcp-server

BEEHIIV MCP

github.com/danvega/beehiiv-mcp-server



spring

THANK YOU!

danvega@gmail.com

https://www.danvega.dev