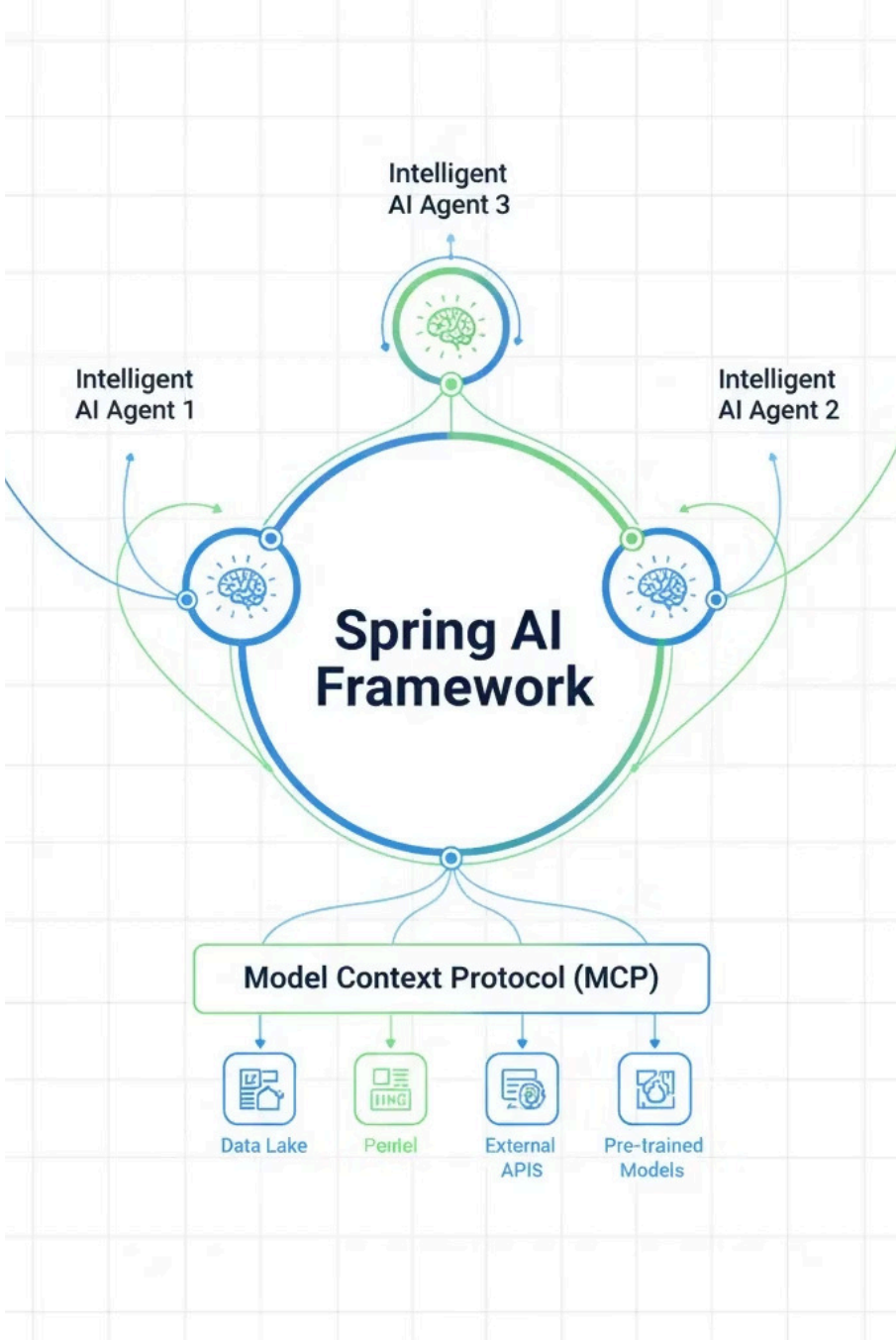
Spring AI 활용 - 지능형 Agent

도구호출과 MCP를 통한 LLM 기능 확장



프로젝트 환경 설정

기본 설정

- 빌드 도구: Gradle - Groovy
- 언어: Java 25
- Spring Boot: 3.5.x
- Spring AI: 1.1.2 (안정화 버전)

Spring Boot 3.5.x를 선택하여 Spring AI와의 호환성을 확보합니다.

필수 의존성

- Spring Web & Reactive Web
- Spring AI (OpenAI)
- JSoup Document Reader
- Thymeleaf
- Lombok

핵심 개념

도구 호출(Tool Calling)이란?

도구 호출은 LLM이 스스로 해결할 수 없는 작업을 위해 애플리케이션의 기능을 빌려 쓰는 메커니즘입니다.

지식의 확장

학습 데이터에 없는 최신 정보나 내부 데이터베이스를 실시간으로 조회합니다.

실행의 주체

실제 코드를 실행하는 것은 LLM이 아니라 애플리케이션입니다. LLM은 어떤 도구를 실행할지만 결정합니다.

보안 및 제어

실행 권한이 애플리케이션에 있어 개발자가 실행 과정을 통제하고 보안을 관리할 수 있습니다.

도구 호출 동작 단계

Chat 요청
사용자 질문과 도구 정의 전송

AI 판단
모델이 도구 필요성 평가

Dispatch 요청
Spring AI가 호출 가로채기

도구 실행
애플리케이션 내부 도구 실행

사용자의 질문이 도구 실행을 거쳐 최종 답변으로 완성되는 전체 흐름입니다.

요청 단계

- 사용자 질문과 도구 정의 전달
- AI Model의 도구 필요성 판단
- Spring AI의 요청 가로채기

실행 및 응답

- 애플리케이션 내부 Tool 실행
- 실행 결과를 AI Model에 전달
- 최종 답변 생성 및 제공

도구 정의하기

Spring AI에서 도구는 @Tool 어노테이션을 적용한 메서드로 정의됩니다.

접근 제어자

public, protected, default, private 모두 가능합니다.

메서드 형태

정적 메서드나 인스턴스 메서드 모두 지원합니다.

클래스 구성

일반 POJO 또는 @Component 빈 내에 정의 가능하며 중첩 클래스도 지원합니다.

주요 어노테이션

@Tool (도구 설정)

- **description**: LLM이 도구를 언제 호출할지 판단하는 근거 (필수)
- **name**: 도구의 이름 (옵션, 미지정 시 메서드명 사용)
- **resultConverter**: 결과를 문자열로 변환하는 클래스
- **returnDirect**: true시 결과를 클라이언트에 즉시 반환

@ToolParam (매개변수 설정)

- **description**: 매개값의 형식과 허용 범위 설명 (필수)
- **required**: 필수 값 여부 (기본값 true)

@Nullable이 적용되고 required가 명시되지 않으면 선택사항으로 간주됩니다.

실습 예제

날짜/시간 도구 구현

현재 시간 조회와 알람 설정 기능을 제공하는 도구를 구현합니다.

01

DateTimeTools 클래스 생성

@Component로 등록하고 @Tool 어노테이션으로 메서드를 정의합니다.

02

getCurrentDateTime 메서드

현재 날짜와 시간 정보를 ISO-8601 형식으로 반환합니다.

03

setAlarm 메서드

지정된 시간에 알람을 설정하며, 24시간 형식 예외를 처리합니다.

04

서비스 통합

ChatClient에 도구를 등록하고 테스트를 수행합니다.

추가 데이터 제공 - **ToolContext**

ToolContext는 LLM을 거치지 않고 애플리케이션에서 도구로 직접 주입되는 데이터 저장소입니다.

보안성

인증 토큰, 세션 ID, API 키 등 민감한 정보를 안전하게 전달합니다.

직접 주입

AI 모델을 거치지 않고 Spring AI 프레임워크가 도구에 직접 데이터를 전달합니다.

유연성

Map 형태로 어떠한 형태의 객체도 담을 수 있습니다.

도구 예외 처리

도구 실행 중 발생하는 예외를 관리하는 ToolExecutionExceptionProcessor를 활용합니다.

기본 처리 방식

DefaultToolExecutionExceptionProcessor가 에러 메시지를 LLM에게 전달합니다.

- LLM이 오류 원인을 설명
- 수정된 파라미터로 재시도
- 흐름이 끊기지 않고 복구

커스텀 처리 방식

throwException을 true로 설정하여 애플리케이션에서 직접 제어합니다.

- LLM을 거치지 않고 즉시 예외 발생
- 프로세스 중단 또는 별도 로직 수행
- 개발자가 완전한 제어권 확보

실무 활용

파일 관리 도구

자연어 인터페이스를 통해 복잡한 CLI 명령어 없이 파일 작업을 수행합니다.

직관성

"보고서 요약해줘", "폴더 생성해줘"와 같은 명령어로 작업 가능합니다.

생산성

반복적인 CRUD 작업을 자동화하여 시간을 절약합니다.

집중도

기술적 조작보다 분석 및 창의적 업무에 리소스를 투입할 수 있습니다.

보안 주의사항: 도구는 반드시 지정된 작업 디렉토리 내에서만 동작해야 하며, System Root 접근을 엄격히 제한해야 합니다.

인터넷 검색 도구

LLM의 Knowledge Cutoff 문제를 해결하기 위해 검색 엔진 API와 연동합니다.

1. **최신성 보완**

   실시간 뉴스, 주가, 날씨, 기술 업데이트 정보를 반영합니다.

2. **신뢰성 확보**

   답변의 근거(URL, 출처)를 제공하여 투명성을 제고합니다.

3. **근거 기반 답변**

   LLM이 도구 결과를 바탕으로 추론하여 할루시네이션을 방지합니다.

Brave Search API, Google Custom Search API 등 다양한 검색 서비스를 활용할 수 있습니다.

MCP

MCP(Model Context Protocol)

MCP는 AI 애플리케이션이 외부의 다양한 도구 및 데이터 소스와 표준화된 방식으로 통신할 수 있게 해주는 오픈 소스 프로토콜입니다.

표준화

MCP 규격만 맞추면 도구마다 새로운 연동 코드를 작성할 필요가 없습니다.

LLM 한계 극복

실시간 검색, 파일 수정, DB 쿼리, IoT 제어 등을 안전하게 수행합니다.

유연한 생태계

Java 외에도 Python, Node.js로 작성된 MCP 서버를 활용할 수 있습니다.

MCP 통신 방식

STDIO (표준 입출력)

로컬 프로세스 간의 직접적인 파이프라인 통신

- 동일 머신 내에서 실행
- 파이프 메모리 버퍼 사용
- 지연 시간 거의 없음
- 별도 네트워크 설정 불필요

SSE (Server-Sent Events)

원격 서버와의 지속적인 HTTP 단방향 푸시

- 독립적인 웹 서버로 구동
- 네트워크를 통한 원격 통신
- 확장성이 뛰어남
- 여러 애플리케이션 공통 사용

MCP Client/Server

클라이언트와 서버 간 역할 정의

MCP Session

연결 세션 관리와 상태 보존

MCP Transport

STDIO 또는 SSE를 통한 전송

정리

핵심 요약

도구 호출의 이해

LLM이 애플리케이션의 기능을 빌려 쓰는 메커니즘으로, @Tool 어노테이션을 통해 정의합니다.

실무 도구 구현

날짜/시간, 파일 관리, 인터넷 검색 등 다양한 실무 도구를 Spring AI로 구현할 수 있습니다.

MCP 프로토콜

표준화된 방식으로 외부 도구와 통신하며, STDIO와 SSE 두 가지 방식을 지원합니다.

예외 처리와 보안

ToolContext를 통한 안전한 데이터 전달과 예외 처리로 신뢰할 수 있는 시스템을 구축합니다.