MCP는 직접 만들어 쓰자 2주차

FastMCP 사용해보기

한양대학교 컴퓨터소프트웨어학부 박철완

오늘 배울 내용

FastMCP로 AI 도구 개발의 첫 걸음

1. MCP(Model Context Protocol)가 무엇인지 이해하기
2. 기존 MCP 개발의 어려움 파악하기
3. FastMCP의 핵심 개념과 장점 학습하기
4. 다국어 Hello World 실습으로 직접 체험하기

Model Context Protocol (MCP) 이해하기

AI와 외부 도구를 연결하는 표준 프로토콜

- MCP = AI가 외부 함수와 데이터를 사용할 수 있게 하는 표준
- Claude, ChatGPT 등이 여러분의 함수를 직접 실행 가능
- 예시: "파일 읽기", "계산하기", "데이터베이스 조회" 등
- AI 생태계에서 MCP의 역할과 중요성

표준 MCP 개발의 현실

간단한 기능도 50줄 이상의 복잡한 코드가 필요

덧셈 함수 하나 만들기 위해 필요한 것들:

- 비동기 서버 설정
- JSON 스키마 수동 작성
- 복잡한 핸들러 함수들

```python
# 표준 MCP - 복잡한 덧셈 서버
import asyncio
from mcp.server import Server
from mcp.server.stdio import stdio_server
from mcp.types import Tool, TextContent

server = Server("my-server")

@server.list_tools()
async def list_tools():
    return [Tool(
        name="add",
        description="Add two numbers",
        inputSchema={
            "type": "object",
            "properties": {
                "a": {"type": "number"},
                "b": {"type": "number"}
            },
            "required": ["a", "b"]
        }
    )]

@server.call_tool()
async def call_tool(name: str, arguments: dict):
    if name == "add":
        result = arguments["a"] + arguments["b"]
        return [TextContent(
            type="text", text=str(result)
        )]

# ... 50줄 이상의 보일러플레이트 코드
```

FastMCP - 모든 것을 단순하게

Python 함수에 데코레이터 하나만 추가하면 AI 도구 완성

FastMCP: "복잡한 건 FastMCP가, 개발자는 로직에만 집중"

```
@mcp.tool() 데코레이터 하나로 완성되는 덧셈 함수

@mcp.tool()
def 더하기(a: int, b: int) -> int:
    return a + b
```

세 가지 데코레이터

@mcp.tool(), @mcp.resource(), @mcp.prompt()로 모든 MCP 연동 해결

@mcp.tool()

특정 기능을 실행하는 "행동 단위"
사용자가 명령을 내리면 실제로 동작을 실행
예시: 텍스트 요약, 코드 실행, 데이터베이스 쿼리

```python
# 함수를 AI가 실행할 수 있는 도구로 변환
@mcp.tool()
def calc(x: int) -> int:
    return x * 2
```

@mcp.resource()

정보를 표현하고 공유하는 "데이터 단위"
예시: 특정 문서, 데이터셋, 파일

```python
# 데이터를 AI에게 제공
@mcp.resource()
def get_data():
    return "데이터"
```

@mcp.prompt()

모델에게 지시를 내리는 "설명/질문" 템플릿
일반적으로 알고 있는 프롬프트와 동일
예시: MCP 사용법 프롬프트

```python
# 프롬프트 템플릿 제공
@mcp.prompt()
def template():
    return "템플릿"
```

개발 환경 준비하기

uv와 FastMCP 설치로 5분 만에 준비 완료

1

uv 설치 (Python 패키지 관리자)

```
# MacOS
curl -LsSf https://astral.sh/uv/install.sh | sh

# Windows
powershell -ExecutionPolicy ByPass -c "irm https://astral.sh/uv/install.ps1 | iex"
```

2

FastMCP 설치

```
uv add fastmcp
```

3

프로젝트 초기화

```
uv init
```

10줄 미만으로 만드는 첫 MCP

파일 하나로 AI가 사용할 수 있는 인사 도구 완성

```python
from fastmcp import FastMCP

mcp = FastMCP("Hello World")

@mcp.tool()
def hello(name: str) -> str:
    """이름을 받아서 인사하는 함수"""
    return f"Hello {name}"

if __name__ == "__main__":
    mcp.run()
```

FastMCP 개발 주의점

타입 힌트, 독스트링, 데코레이터 반드시 기억하기

```
# 타입 힌트 필수

✅ def func(name: str) -> str

❌ def func(name)

# 함수 설명(docstring) 작성

def foo():
    ✅ """이 함수는 무엇을 합니다"""

    ❌ # 설명 없음

# mcp.tool 데코레이터

✅ @mcp.tool()

❌ 데코레이터 없음
```

함께 만들어보는 다국어 인사 도구

6개 언어로 인사하는 AI 도구 제작

🇰🇷
한국어
"안녕하세요"

🇺🇸
영어
"Hello"

🇪🇸
스페인어
"Hola"

🇯🇵
일본어
"こんにちは"

🇩🇪
독일어
"Hallo"

🇫🇷
프랑스어
"Bonjour"

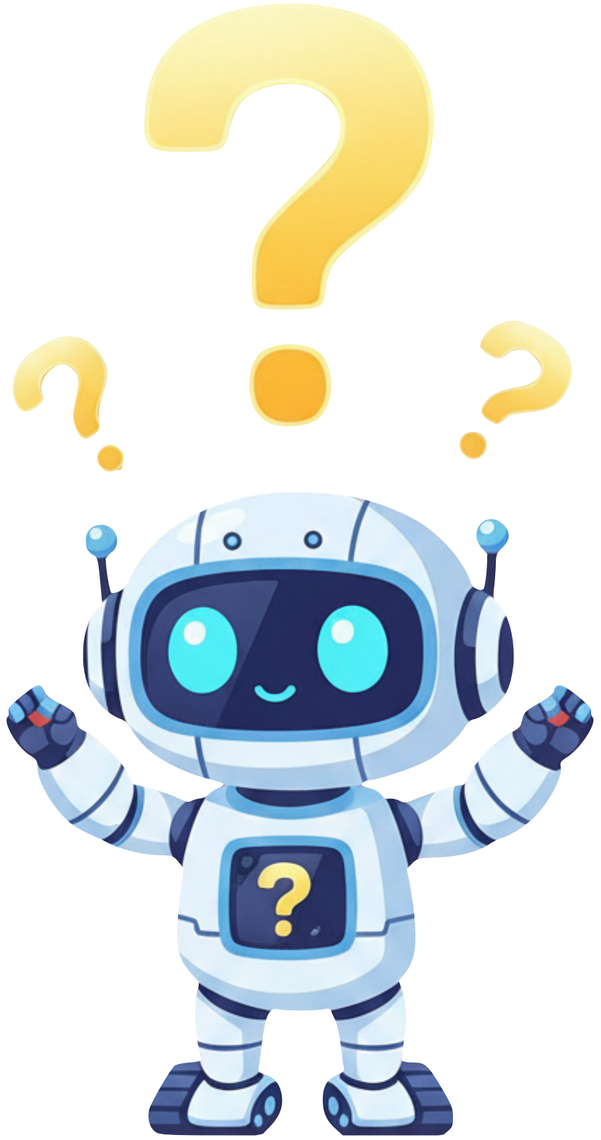
각 언어별로 개별 함수 작성, 단일 파일로 구성

FastMCP로 만들 수 있는 것들

간단한 파일입출력, 데이터 분석, 웹 크롤링 뿐 아니라
여러분의 아이디어가 AI 도구가 될 수 있습니다.

궁금한 것들을 해결해 봅시다

Q&A



질문 예시

- "FastMCP vs 표준 MCP 성능 차이는?"
- "실제 상용 서비스에서 사용 가능한가요?"
- "다른 프로그래밍 언어는 지원하나요?"