



MCP 입문 강의 1주차

Model Context Protocol

강사 소개



박철완

- 한양대 CS 4학년 (2020.03-)
- 현재 Hyperithm IA셀 인턴 중 (2025.07-)
- 공군 기상소프트웨어개발병: 공군기상정보시스템/국방기상지원체계 개발 (2022.04-2024.01)
- Transverse에서 LMS Backend 개발 (2020.11-2021.10)

LLM 활용 방식 변화

LLM은 단순 텍스트 생성을 넘어 활용 방식이 진화하고 있습니다. 툴 콜링으로 외부 도구를 활용하고, 컨텍스트로 지식을 확장하며, 에이전트 기술로 자율적인 문제 해결이 가능해졌습니다. 이는 AI의 활용 범위와 효율성을 크게 확장시켰습니다.



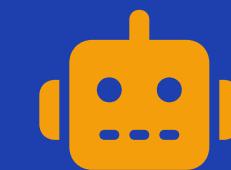
툴 콜링

AI가 외부 도구를 호출하고 활용하는 기술로, 실시간 정보 검색, 데이터 분석, API 활용 등 AI의 능력을 확장합니다. 최신 모델들은 필요한 도구를 스스로 선택하고 활용할 수 있습니다.



컨텍스트

AI에게 관련 정보를 제공하는 기술로, 문서 검색, 지식베이스 연결, 대량의 데이터 처리를 가능하게 합니다. RAG 기술을 통해 할루시네이션을 줄이고 정확도를 크게 향상시켰습니다.



에이전트

AI가 목표를 설정하고 계획을 세워 자율적으로 문제를 해결하는 기술입니다. 코드 작성, 복잡한 작업 자동화, 지속적인 학습 및 적응이 가능하며 인간의 개입 없이도 작업을 완수합니다.

툴 콜링이란?

툴 콜링(Tool Calling)은 AI 모델이 외부 도구나 API를 **자율적으로 호출**하여 정보를 얻거나 작업을 수행하는 기능입니다. 이를 통해 AI는 본래 학습하지 않았거나 실시간 정보가 필요한 질문에도 답변할 수 있게 됩니다.



AI가 도구를 **자율적으로 선택**하여 사용합니다



외부 시스템과의 **연결성**을 크게 향상시킵니다

Before



→ "날씨?" →  →  모름

AI는 최신 날씨 정보에 접근할 수 없어 응답 불가

After



→ "날씨?" →  →  → **15°C 맑음**

AI가 날씨 API를 자동으로 호출하여 정확한 정보 제공



활용 사례

웹 검색, 데이터베이스 조회, 파일 시스템 접근, 외부 API 연동

툴 콜링의 장점



확장성

툴 콜링을 통해 LLM의 기능을 무한히 확장할 수 있습니다. 새로운 기능이나 데이터 소스가 필요할 때마다 새로운 도구를 연결하기만 하면 됩니다. 이는 AI 시스템의 수명과 활용 범위를 크게 확장합니다.

툴 콜링은 AI 모델이 외부 시스템과 도구에 접근할 수 있게 하는 핵심 기술입니다. 이를 통해 실시간 데이터 접근, API 호출, 파일 시스템 조작 등이 가능해져 AI의 활용 범위가 크게 확장됩니다. 플랫폼 별로 다양한 구현 방식이 존재하지만 모두 동일한 목적을 가지고 있습니다.



안전성

외부 시스템과의 통신을 구조화된 방식으로 관리함으로써 보안과 권한 관리가 용이해집니다. 각 도구는 명확히 정의된 입력 인터페이스를 통해 안전하게 상호작용합니다.



효율성

LLM이 직접 정보를 생성하는 대신 외부 시스템에서 정확한 정보를 가져오므로 할루시네이션이 감소하고 응답의 정확도가 향상됩니다. 또한 최신 데이터에 실시간으로 접근할 수 있습니다.

컨텍스트 엔지니어링 - RAG



RAG(Retrieval-Augmented Generation)는 검색 기능과 생성 AI를 결합하여 정확하고 최신 정보를 제공하는 파이프라인입니다. 대용량 문서 데이터베이스에서 관련 정보를 검색하고 LLM 프롬프트에 추가함으로써 할루시네이션을 줄이고 응답 품질을 향상시킵니다. LangChain → Perplexity → NotebookLM 등의 도구로 구현됩니다.

질문 분석



사용자 질문을 분석하여 의도를 파악하고, 검색에 필요한 핵심 키워드와 개념을 추출합니다. 복잡한 질문은 여러 하위 질문으로 분해하여 효과적인 검색을 준비합니다.

컨텍스트 검색 및 보강



벡터 데이터베이스에서 의미적으로 유사한 문서를 검색하고, 관련성과 신뢰도를 평가합니다. 검색된 정보를 사용자 질문과 결합하여 LLM에게 풍부한 컨텍스트를 제공합니다.

응답 생성 및 최적화



검색된 정보를 바탕으로 정확하고 신뢰할 수 있는 응답을 생성합니다. 인용 출처를 명시하고, 확실하지 않은 정보에 대해서는 불확실성을 표현하며, 사용자 의도에 최적화된 형태로 답변을 구조화합니다.

자율 AI 에이전트



에이전트 작동 원리

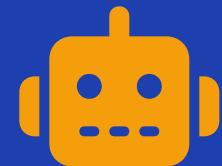
에이전트는 목표 설정에서 시작해 계획을 수립하고, 실행 후 결과를 평가하여 다시 목표를 조정하는 순환적 과정을 반복합니다. 이 반복적인 피드백 루프를 통해 자율적으로 의사결정을 하며 복잡한 작업도 점진적으로 해결할 수 있습니다.

자율 AI 에이전트는 사용자 목표를 위해 스스로 판단하고 행동하는 인공지능 시스템입니다. 목표와 피드백을 바탕으로 자율적으로 계획을 세우고 실행하며, 결과를 평가하는 순환적 프로세스를 통해 연속적으로 작업을 수행합니다.



일반 AI

- 단일 질문에 대한 단일 응답만 제공 • 맥락에 대한 기억이 제한적
- 사용자 지시에만 수동적으로 반응 • 외부 도구 활용 능력 제한적
- 일회성 상호작용에 최적화



에이전트

- 연속적인 작업 수행 가능
- 자율적인 판단과 의사결정
- 중간 결과를 평가하고 계획 조정
- 다양한 도구를 스스로 선택 활용
- 장기적 목표 달성을 최적화

에이전트 세대별 발전

AI 에이전트 기술은 급격한 발전을 이루었습니다. 초기의 단순 반복 실행 모델에서 시작하여, 환경 이해와 자율성을 갖춘 에이전트로 진화했으며, 현재는 복잡한 코드베이스를 이해하고 전문가 수준의 작업을 수행할 수 있는 단계로 발전했습니다.



2023

AutoGPT / BabyAGI

-  반복 실행 방식의 초기 에이전트
-  무한 루프 문제 발생
-  간단한 작업 자동화에 제한적 활용



2024

Devin / Computer Use

-  개발 환경에 대한 이해 증가
-  장시간 자율 작업 가능
-  복잡한 개발 과제 해결 능력



2025

Cursor / Claude Code

-  대규모 코드베이스 이해 가능
-  전문가급 소프트웨어 엔지니어 능력 달성
-  프로젝트 관리 및 최적화 역량 확보

초기 실험 단계에서 전문가급 성능까지 **매우 빠른 시간 내에 혁신적 발전**을 이루었습니다

공통점

툴 콜링, 컨텍스트 엔지니어링, AI 에이전트의 세 가지 LLM 핵심 기술은 서로 다른 영역을 다루지만, 실제 구현에 있어서는 공통적인 요구사항을 가지고 있습니다. 이러한 공통점이 MCP와 같은 표준화된 프로토콜의 필요성을 더욱 부각시킵니다.



외부 데이터 접근

다양한 소스와 형식의 외부 데이터에 안정적으로 접근하고 처리할 수 있는 능력이 필요합니다. 파일, API, 데이터베이스 등 여러 데이터 소스를 통합적으로 활용해야 합니다.



실시간 처리

사용자 요구와 환경 변화에 신속하게 대응하기 위한 실시간 처리 능력이 요구됩니다. 대기 시간을 최소화하고 정보의 최신성을 보장해야 합니다.



다양한 형식 지원

텍스트, 이미지, 코드, 구조화된 데이터 등 다양한 형식의 정보를 처리하고 변환할 수 있어야 합니다. 멀티모달 처리 능력이 점점 더 중요해지고 있습니다.



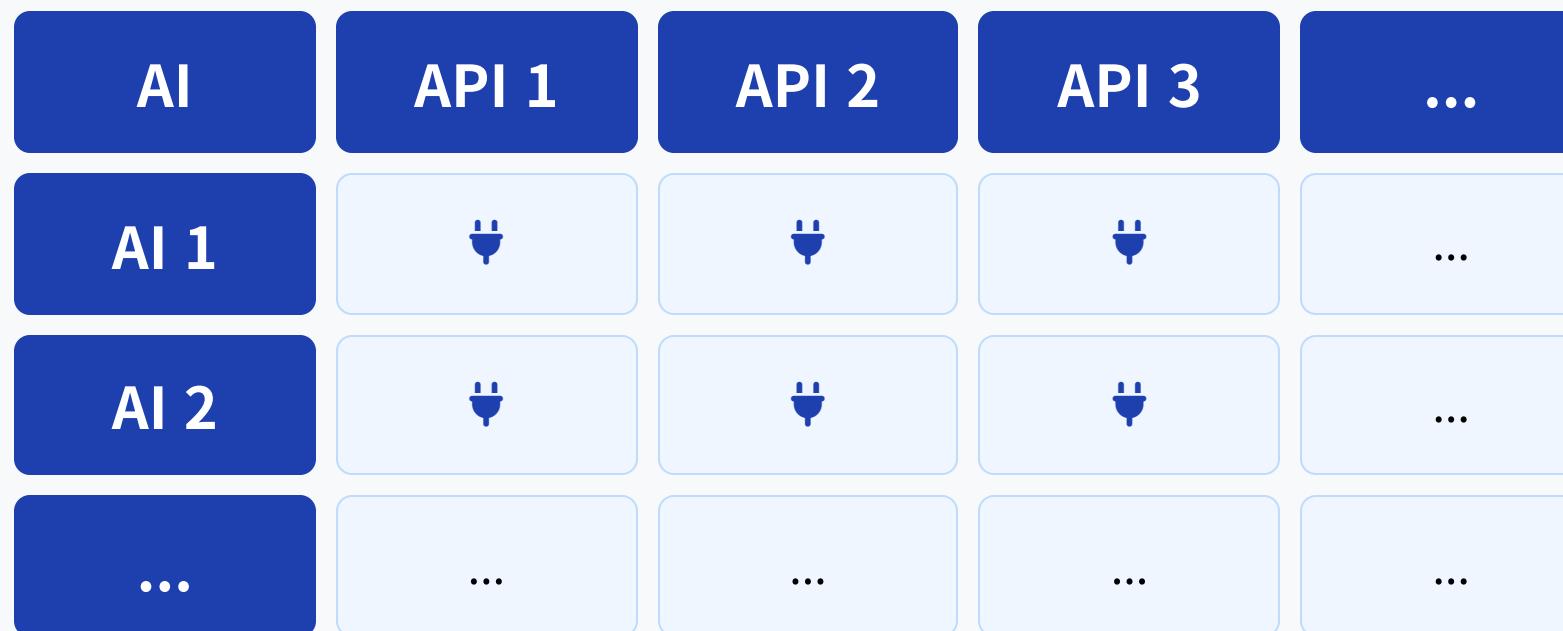
보안/인증

민감한 데이터와 시스템에 대한 접근을 관리하고, 권한 부여 및 인증 메커니즘을 통해 시스템의 안전을 보장해야 합니다. 규정 준수와 데이터 보호가 필수적입니다.

문제점

N×M 문제

다수 AI × 다수 API=많은 커넥터



AI 모델과 외부 시스템 간의 연결 방식이 표준화되지 않아, 모델별로 다른 인터페이스를 구현해야 합니다.

개발/유지보수 비용

</> 개발: 모든 커넥터를 개발해야함

⚙️ 유지보수: 모든 커넥터 유지보수 필요

💰 인건비: 개발/유지보수에 비용 발생

✗ 개발 중복: 같은 기능을 여러 모델에 반복 구현

✗ 유지보수 부담: API 변경 시 모든 커넥터 수정 필요

✗ 높은 비용: 다수 AI × 다수 서비스 = 많은 구현 = 높은 비용

⚠️ 표준 필요!

개발자 고통

N × M 지옥

AI 모델과 서비스를 연동할 때마다 커넥터를 개별적으로 구현해야 하는 개발자 고통을 정리하면 다음과 같습니다.

다수 AI × 다수 서비스 = **많은 연동**
= 장기간 개발 시간
= **높은 비용** 발생

"같은 코드를 여러 번 작성해야 합니다."

"모델이 업데이트될 때마다 코드를 재작성해야 합니다."



⚠ 표준 필요!

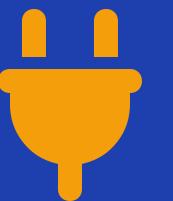
MCP 핵심

AI의 USB



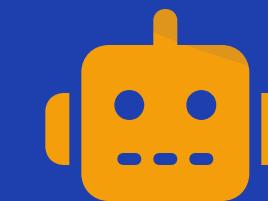
Before: 다양한 포트

각 장치마다 다른 연결 방식
호환성 문제와 비효율



USB: 하나의 표준

통일된 인터페이스
장치 간 호환성 확보



MCP: AI 표준

AI 모델-외부세계 연결 표준화
개발 효율성 극대화

MCP 아키텍처



MCP는 AI 모델이 외부 시스템과 표준화된 방식으로 통신할 수 있게 해주는 프로토콜입니다. JSON-RPC 기반으로 다양한 AI 모델과 서비스 간의 연결을 단순화합니다.

Transport 레이어

통신 프로토콜과 데이터 전송을 담당합니다. HTTP, WebSocket 등의 프로토콜을 통해 AI 모델과 서버 간의 안정적인 데이터 전송을 보장합니다. 다양한 환경과 네트워크 조건에서 호환성을 유지합니다.

Message 레이어

JSON-RPC 형식의 메시지 구조를 정의합니다. 명확한 method와 params 규격으로 요청과 응답을 표준화하며, 에러 처리와 비동기 통신을 지원합니다. 일관된 인터페이스로 개발자 경험을 향상시킵니다.

Protocol 레이어

AI와 서버 간의 상호작용 규칙을 정의합니다. 파일 시스템 접근, 데이터베이스 쿼리, API 호출 등의 표준 작업을 위한 프로토콜 스펙을 제공하여 다양한 서비스와의 통합을 단순화합니다.

LLM의 감각

MCP는 LLM에게 외부 세계를 인지할 수 있는 감각기관 역할을 합니다.
파일 시스템, API, 이벤트 3가지 인터페이스를 통해 실시간으로 환경과
상호작용할 수 있게 해줍니다.



파일 시스템

코드베이스, 문서, 이미지 등 로컬 파일이나 원격 스토리지에 접근하여 데이터를 읽고 쓸 수 있습니다.



API

외부 서비스와 상호작용하여 실시간 데이터를 가져오거나 작업을 수행할 수 있습니다.



이벤트

시스템 변경사항을 감지하고 실시간으로 반응하여 지속적인 모니터링이 가능합니다.

개발자 친화적

MCP는 개발자 경험을 최우선으로 설계되었습니다. 익숙한 웹 서버 개발 패턴을 그대로 사용하며, 복잡한 AI 통합 작업을 간소화하여 개발자가 AI 기능에 집중할 수 있도록 도와줍니다.



익숙한 패턴

Express.js와 같은 웹 서버 개발 방식을 그대로 적용할 수 있어 학습 곡선이 매우 낮습니다. 미들웨어, 라우팅 등 웹 개발자에게 익숙한 패턴을 그대로 활용합니다.



손쉬운 연동

다양한 AI 모델과 쉽게 연동할 수 있으며, 모델 변경 시 코드 수정이 최소화됩니다. OpenAI, Claude, Gemini 등 여러 모델을 동일한 인터페이스로 사용할 수 있습니다.



빠른 프로토타이핑

최소한의 코드로 AI 기능을 구현할 수 있어 프로토타입 개발 시간이 크게 단축됩니다. 몇 줄의 코드만으로 복잡한 AI 애플리케이션을 빠르게 만들 수 있습니다.

```
const mcp = require('@mcpjs/core');
const app = mcp();

app.use(mcp.router);
app.listen(3000);
```

 5분 만에 시작 가능

 npm install로 간단한 설치

 모듈식 구조로 유연한 확장

3부 - 실습 소개



설치

MCP 서버 설치 및 기본 환경 구성

- ✓ Claude Desktop 설치
- ✓ Gemini-CLI 설치



Claude Desktop

Claude Desktop을 MCP와 연결

- ✓ 파일시스템 접근해보기
- ✓ Gmail 접근해보기



Gemini-CLI

터미널에서 사용하는 Gemini 도구

- ✓ context7 연결해보기
- ✓ playwright 연결해보기

준비사항

모든 실습은 Node.js 18+ 환경에서 진행되며, VS Code와 같은 텍스트 에디터와 터미널이 필요합니다. 각 도구별 API 키는 강의 중 안내됩니다.

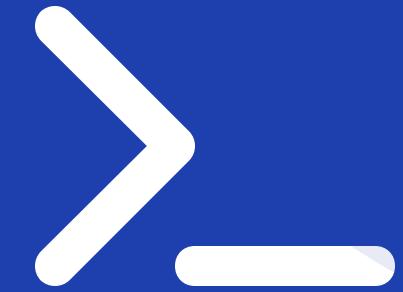
설치

MCP 실습을 위해 필요한 도구를 설치하는 방법을 알아봅니다. Claude Desktop과 Gemini-CLI 설치를 통해 MCP의 다양한 기능을 활용해 보겠습니다.



Claude Desktop 설치

- 공식 웹사이트에서 다운로드
- 설치 파일 실행 및 가이드 따라하기
- API 키 설정 및 연동



Gemini-CLI 설치

- npm install -g @gemini-ai/cli
- API 키 환경변수 설정
- 명령어로 정상 설치 확인



실습 중 문제가 발생하면 언제든지 질문해주세요!

Claude Desktop

MCP를 통해 Claude Desktop을 활용하는
두 가지 핵심 실습을 진행합니다.



파일시스템 접근

Claude Desktop이 로컬 파일과 폴더
에 접근하여 코드, 문서, 이미지 등의
파일을 분석하고 처리하는 방법을 실습
합니다.



Gmail 접근

Claude Desktop이 Gmail 계정에 연
결하여 이메일 데이터를 검색하고 분석
하는 방법을 실습합니다.



실습 과정에서는 직접 Claude Desktop을 설치하고 MCP와 연동하여 활용하게 됩니다.

Gemini-CLI



context7 연결해보기

다양한 소스에서 컨텍스트를 수집하고
AI에게 제공하는 확장 도구



playwright 연결해보기

웹 브라우저 자동화를 통해
웹 데이터를 수집하고 처리

오늘의 요약

오늘 배운 내용을 크게 세 가지로 요약할 수 있습니다. LLM의 최신 발전 동향과 MCP의 핵심 가치, 그리고 실습을 통해 확인한 효과입니다. 이 세 가지 핵심을 이해하면 현대 AI 개발의 방향성과 MCP가 해결하는 문제를 명확히 파악할 수 있습니다.

오늘 학습한 내용: 핵심 개념 **다수**, 실용 도구 **여러개**

AI 진화 → 외부 연결

LLM의 3대 핵심 진화(툴 콜링, 컨텍스트, 에이전트)는 모두 외부 시스템과의 연결이 핵심입니다. AI는 더 이상 폐쇄된 모델이 아니라 외부 데이터, API, 도구와 상호 작용하며 능력을 확장하고 있습니다. 이러한 발전은 AI가 단순 응답을 넘어 실제 작업을 수행할 수 있게 합니다.

MCP → N × M을 N+M으로

MCP는 N × M 문제를 해결하는 표준화 프로토콜입니다. 다수의 AI 모델과 서비스를 연결할 때, 기존 방식보다 훨씬 적은 수의 커넥터만 필요합니다. 이를 통해 개발 시간을 대폭 단축하고 유지보수 비용을 크게 절감할 수 있습니다.

실습

오늘의 실습을 통해 MCP의 효과를 직접 확인했습니다. 파일 탐색은 크게 개선되었고, 자동화는 현저히 향상되었습니다. 정확도 또한 확연히 증대되었습니다. 개발자는 상당한 시간과 비용을 절감할 수 있으며, 반복 작업 대신 고부가가치 개발에 집중할 수 있습니다.

다음 주

Week 2 예고

다음 주에는 **MCP 서버를 직접 만들어** 볼 예정입니다. MCP의 내부 구조를 이해하고, 직접 서버를 구현하면서 실제 운영 환경에서 활용할 수 있는 실무 역량을 키우게 됩니다.

준비사항

🐍 최신 Python

</> VS Code

⬇️ FastMCP 패키지 사전 설치

코드 미리보기: FastMCP

```
from fastapi import FastAPI
from fastmcp import MCPServer

app = FastAPI()
mcp = MCPServer()

@mcp.tool("get_weather")
async def get_weather(city: str):
    """도시의 날씨 정보를 반환합니다."""
    # API 연동 코드
    return {"temp": "적정", "condition": "맑음"}


@app.post("/mcp")
async def handle_mcp(request: dict):
    return await mcp.handle_request(request)

# 간단하게 완성되는 MCP 서버!
```

학습 목표

- MCP 서버 아키텍처 이해하기
- FastAPI와 FastMCP를 활용한 서버 구축
- 다양한 도구 연결 방법 학습
- 실제 AI 모델과 연동 및 테스트

질문?

-  Email
-  Discord
-  GitHub

참고 자료

-  MCP 문서
-  FastMCP
-  강의자료

▣ 과제

Context7 앱 만들기