Guida alla Creazione di un MCP Server per Generazione Test

Luigi Cirillo

Gennaio 2026

1 Introduzione: Cos'è un MCP Server?

L'MCP (**Model Context Protocol**) è un protocollo progettato per fornire un contesto ricco e strutturato a un LLM (Large Language Model). Lo scopo principale è permettere all'AI di interagire direttamente con l'ambiente di sviluppo, evitando colli di bottiglia, riducendo il consumo di token e ottimizzando le risorse.

Un MCP Server permette all'LLM di:

- Connettersi al progetto di lavoro come un "Super User".
- Conoscere l'intera struttura delle dipendenze.
- Sviluppare codice pulito e limitare gli errori seguendo i pattern esistenti.

Nota importante: La generazione automatica non è infallibile. Lo sviluppatore umano rimane il supervisore finale che deve validare ogni modifica.

2 Prerequisiti e Setup

Per questa guida utilizzeremo **Python** e il moderno package manager **uv**. Assicurati di avere installato:

1. Python 3.10+
2. Pip
3. **uv** (consigliato per la gestione rapida di dipendenze e venv).

3 Inizializzazione del Progetto

Utilizzando il terminale (PowerShell su Windows 11), posizionati nella directory di sviluppo e lancia i seguenti comandi:

```
# Inizializza il progetto e crea il virtual environment
uv init mcp-tests-tools
cd mcp-tests-tools

# Aggiunge le dipendenze necessarie
```

```
uv add mcp
uv add beautifulsoup4
```

Nel caso in cui si ha a disposizione il codice sorgente con i file .toml e del file .lock, aprire la directory dove presente il sorgente è lanciare da terminale il comando:

```
# Permette la creazione dell'ambiente usando le dipendenze gia
   importate nel progetto
uv sync
```

4 Struttura del Progetto

Per una logica pulita, creiamo una cartella `src` per ospitare i nostri script:

```
mkdir src
cd src
new-item main.py #per la creazione di nuovi script python
```

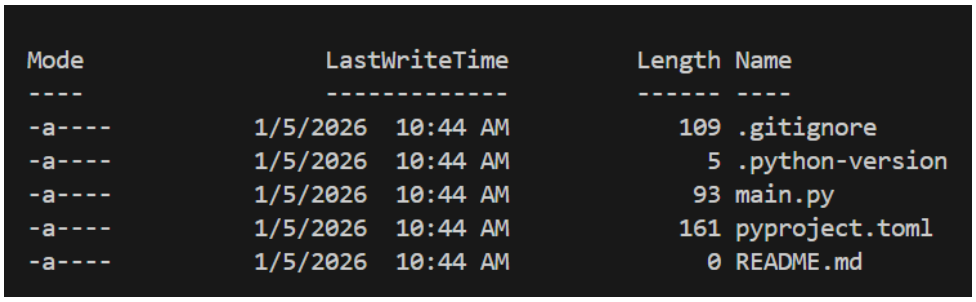

Figura 1: Rappresentazione della struttura delle cartelle del progetto.

5 Le Annotation MCP

Il cuore del server è definito da tre decoratori principali:

- `@mcp.tool()`: Definisce un'azione eseguibile (es. analizzare un file, scrivere codice).
- `@mcp.resource()`: Definisce l'accesso a dati statici (es. metadati, file pom.xml).
- `@mcp.prompt()`: Definisce template per guidare l'AI verso uno stile specifico.

6 Testing e Debugging

Per verificare il funzionamento dei metodi creati, utilizziamo l'inspector ufficiale:

```
npx @modelcontextprotocol/inspector uv run src/main.py
```

PS: Dopo aver aperto l'interfaccia web dell mcp, avviare la connessione e successivamente accedere alla sezione tool dove ci saranno tutti i metodi implementati. Testare per capire se risponde nella maniera giusta.

7 Configurazione Claude Desktop

Per rendere i tool disponibili su Claude Desktop, è necessario configurare il file JSON di sistema.

Il comando rapido per aprire la configurazione su windows è:

```
code $env:AppData\Claude\claude_desktop_config.json
```

Figura 2: Rappresentazione della struttura delle cartelle del progetto.