MCP fatto bene ◆

Collegare Claude Code a database, issue tracker, API e servizi esterni con il Model Context Protocol: transport, scope, OAuth, timeout e sicurezza.

Aggiornato: luglio 2026 · Fonte: documentazione ufficiale Anthropic (code.claude.com/docs) · Vol. 7 della serie bignami Claude Code.

1. Cos'è e a cosa serve

smettere di copiare dati in chat

MCP (Model Context Protocol) è lo standard open source per le integrazioni AI↔tool. Un server MCP dà a Claude **tool e dati** di un sistema esterno: il segnale che te ne serve uno è quando ti accorgi di copiare in chat contenuti da un'altra finestra — issue tracker, dashboard di monitoring, database. Una volta connesso, Claude legge e agisce direttamente. Esempi dal mondo reale: "implementa la feature dell'issue JIRA ENG-4521 e apri una PR su GitHub", "controlla su Sentry l'uso della feature", "trova nel PostgreSQL le email di 10 utenti che l'hanno usata", "aggiorna il template email dai nuovi design Figma".

Prima regola: fidati, poi connetti

Un server MCP è codice (o un servizio) con accesso alla tua sessione. Server che fetchano contenuti esterni ti espongono al rischio di **prompt injection**. Parti dai connettori revisionati nella **Anthropic Directory** (claude.ai/directory) e ispeziona quelli di community prima di aggiungerli. In azienda, gli admin possono imporre allowlist/denylist di server via managed configuration.

2. Installare i server

tre transport, un doppio trattino cruciale

```
# HTTP remoto (raccomandato per i cloud)
claude mcp add --transport http notion \
  https://mcp.notion.com/mcp

# con Bearer token
claude mcp add --transport http secure-api \
  https://api.example.com/mcp \
  --header "Authorization: Bearer TOKEN"

# stdio locale (processi sulla tua macchina)
claude mcp add --env AIRTABLE_API_KEY=KEY \
  --transport stdio airtable \
  -- npx -y airtable-mcp-server

# SSE: deprecato, usa HTTP dove possibile
# WebSocket (server che pushano eventi):
# via add-json con "type":"ws"
```

Il -- non è decorativo

Per i server stdio, il doppio trattino separa le opzioni di Claude (`--transport`, `--env`, `--scope`) dal comando del server: tutto ciò che segue passa intatto. Senza, Claude Code proverebbe a interpretare i flag del server (tipo `--port`) come propri.

Gestione quotidiana

- `claude mcp list / get / remove` da shell; `/mcp` in sessione mostra stato, conteggio tool e gestisce l'**OAuth** dei server remoti.
- `claude mcp login <nome>` esegue l'OAuth da CLI; `--no-browser` via SSH stampa l'URL da aprire altrove.
- Riconnessione automatica per HTTP/SSE con backoff esponenziale (5 tentativi); gli stdio sono processi locali e non vengono riconnessi.
- I server possono aggiornare i propri tool a caldo (notifiche `list_changed`).
- **Canali**: un server può anche *pushare* messaggi nella sessione (risultati CI, alert, chat) così Claude reagisce mentre non ci sei — opt-in con `--channels`.

◆ Gli scope: dove vive la configurazione

Scope	Carica in	Condiviso col team	Salvato in
`local`	Solo il progetto corrente	No (default)	`~/.claude.json`, sotto il path del progetto
`project`	Solo il progetto corrente	**Sì, via git**	`.mcp.json` nella root del repo
`user`	Tutti i tuoi progetti	No	`~/.claude.json`

Si sceglie con `--scope`. I server da `.mcp.json` di un repo appena clonato restano **in attesa di approvazione** finché non ti fidi del workspace: un repository non può auto-approvarsi i propri server. Usa `local` per credenziali che non devono finire in version control, `project` per lo standard di team, `user` per i tuoi strumenti trasversali.

3. Operativa: timeout, contesto, permessi

i dettagli che fanno la differenza

Manopola	Effetto
`MCP_TIMEOUT=10000 claude`	Timeout di avvio dei server (ms).
`"timeout": 600000 (in .mcp.json)`	Limite wall-clock **per singola chiamata tool** di quel server (min 1000 ms); override di MCP_TOOL_TIMEOUT.
`MAX_MCP_OUTPUT_TOKENS=50000`	Alza il limite oltre cui l'output di un tool MCP genera warning (default 10.000 token).
`CLAUDE_CODE_MCP_TOOL_IDLE_TIMEOUT`	Le chiamate a server remoti che restano mute 5 minuti vengono abortite; qui cambi la finestra (0 = disattiva).
`Tool search (default attivo)`	Le definizioni dei tool MCP sono **deferite**: nel contesto entrano solo i nomi finché un tool non serve davvero. È ciò che rende sostenibili i server con decine di tool. Il set di nomi deve comunque restare stabile per non invalidare la cache.

MCP nei permessi e nei subagenti

- Nome canonico dei tool: `mcp__<server>__<tool>` (per i server dei plugin: `mcp__plugin_<plugin>_<server>__<tool>`).
- Regole: `mcp__github` in deny toglie tutti i tool di quel server; `mcp__*` tutti gli MCP. Gli allow con glob solo dopo il prefisso letterale del server: `mcp__github__get_*`.
- Nei subagenti: campo `mcpServers` per dare a un worker un server specifico (per nome già configurato o definizione inline); `disallowedTools: mcp__github` per toglierglielo.
- Connetti gli MCP **a inizio sessione**: connessioni/disconnessioni a metà invalidano la cache dei prompt (vol. 3).

Costruirne uno tuo

Il plugin ufficiale scaffolda il grosso:

```
/plugin marketplace add \
  anthropics/claude-plugins-official
/plugin install \
  mcp-server-dev@claude-plugins-official
/mcp-server-dev:build-mcp-server
```

Claude ti chiede il caso d'uso e genera un server HTTP remoto o stdio locale. Per i server stdio, Claude Code imposta `CLAUDE_PROJECT_DIR` nell'ambiente del processo: risolvi i path di progetto senza dipendere dalla working directory. I plugin possono infine **impacchettare** server MCP: si connettono da soli quando il plugin è abilitato — il modo più pulito per distribuirli a un team.

▶ MCP + skill: la coppia vincente

L'MCP dà a Claude i **tool** (la connessione al database, l'API di Slack); una skill gli insegna a **usarli bene** (lo schema del vostro DB e i pattern di query del team, le regole di formattazione dei messaggi). Tool senza conoscenza d'uso = tentativi; conoscenza senza tool = teoria. Insieme, e magari impacchettati in un plugin, diventano l'integrazione che si installa in un comando.

Fonti: code.claude.com/docs — mcp, mcp-quickstart, managed-mcp, channels · Directory connettori: claude.ai/directory · Standard: modelcontextprotocol.io