



第32回 「Introduction to MCP」
Anthropic 公式 Claude Academy

Claude Code

MCP とは何か

公式の教材を、順番に

原石工房

隠れた才能に気づき、花開く

この資料について



原石工房

Anthropic とは関係のない、非公式の資料です

公式の学習サイト Claude Academy の日本語訳です

原文（英語）Introduction to Model Context Protocol /
Introducing MCP

今回から5つめのコースに入ります。全9回のうちの第1回です

全51回の、今日はここ



原石工房

Claude Code 101

1 2 3 4 5 6 7 8 9 10 11 12

第1回～第12回

Claude Code in Action

13 14 15 16 17 18 19 20 21

第13回～第21回

スキル

22 23 24 25 26 27

第22回～第27回

手分け (subagents)

28 29 30 31

第28回～第31回

MCP 入門

32 33 34 35 36 37 38 39 40

第32回～第40回

MCP 応用

41 42 43 44 45 46 47 48 49 50 51

第41回～第51回

今日はここ 第32回

今日のレッスン



原石工房

Introduction to MCP の第1回

原題 Introducing MCP

訳 MCP とは何か

面倒なつなぎこみを書かなくても、Claude に前後の事情と道具を渡せるようにする層です

① 解いている困りごと



原石工房

原題 The Problem MCP Solves

「自分の置き場所ぜんぶで、通っていない願い出は？」

MCP が無いと、自分のところで全部こしらえる

get_repos 決め書きと中身を、自分で

search_issues 決め書きと中身を、自分で

list_prs 決め書きと中身を、自分で

get_projects 決め書きと中身を、自分で

GitHub には、ものすごい量の働きがあります

置き場所／送り込みの願い出／困りごとの書き込み／まとめの板…

書いて、試して、手入れし続けるのは、自分

手間も、続く重荷も、大変なものになります

MCP は、その重荷をずらす

道具の決め方と、動かす仕事を、専用のサーバーへ移します

自分で全部こしらえない

「自分の置き場所ぜんぶで、通っていない願い出は？」に応えたいとします

GitHub には、ものすごい量の働きがあります

MCP が無ければ、信じられない数の道具を、自分で作ることになります

① 重荷は、続く



原石工房

「自分の置き場所ぜんぶで、通っていない願い出は？」

MCP が無いと、自分のところで全部こしらえる

get_repos 決め書きと中身を、自分で

search_issues 決め書きと中身を、自分で

list_prs 決め書きと中身を、自分で

get_projects 決め書きと中身を、自分で

GitHub には、ものすごい量の働きがあります

置き場所／送り込みの願い出／困りごとの書き込み／まとめの板…

書いて、試して、手入れし続けるのは、自分

手間も、続く重荷も、大変なものになります

MCP は、その重荷をずらす

道具の決め方と、動かす仕事を、専用のサーバーへ移します

そのつなぎこみを、書いて、試して、手入れし続けるのは自分です

手間も、続く重荷も、大変なものになります

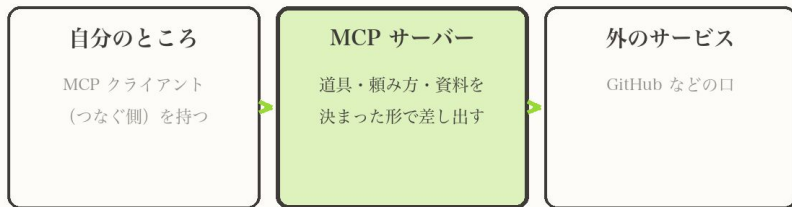
MCP は、道具の決め方と動かす仕事を、専用のサーバーへ移します

自分で全部こしらえない

② どう働くのか

原題 How MCP Works

あいだに、専門の窓口を置く



MCP サーバーは、外のサービスへの窓口としてふるまいます

GitHub の例 サーバーの中に `get_repos()` が入っていて
GitHub の口へ直接つながっています

自分のところはサーバーと話すだけ

GitHub ならではの細かい作りは、サーバーが引き受けます

一から全部こしらえる代わりに、つなぐ

GitHub の道具を全部自分で書く代わりに、GitHub の MCP サーバーが引き受けます

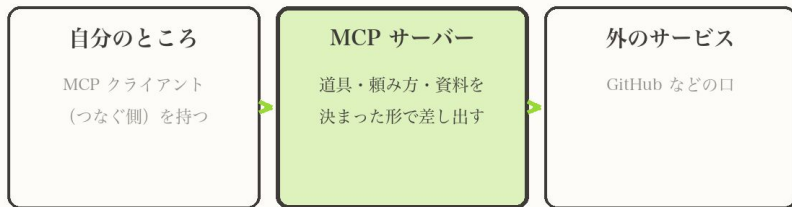
サーバーは、大量の働きを包んで、決まった形の道具として差し出します

自分の作っているものは、一から全部こしらえる代わりに、そこへつながります

③ MCP サーバーとは

原題 MCP Servers Explained

あいだに、専門の窓口を置く



MCP サーバーは、外のサービスへの窓口としてふるまいます

GitHub の例 サーバーの中に `get_repos()` が入っていて
GitHub の口へ直接つながっています

自分のところはサーバーと話すだけ

GitHub ならではの細かい作りは、サーバーが引き受けます

一から全部こしらえる代わりに、つなぐ

外のサービスが持っているデータや働きへの、入口を渡します

道具・頼み方・資料を、決まった形で差し出す、専門の窓口です

GitHub ならではの細かい作りは、サーバーが引き受けます



自分のところ

MCP クライアント（つなぐ側）

使う人の問いを受け、Claude と話す



MCP サーバー

道具・頼み方・資料

外のサービスへの、決まった形の窓口



外のサービス

GitHub・AWS など

実際のデータと働き

道具を作る仕事が、まんなかへ移る

④ よくある問い



原石工房

原題 Common Questions

よくある問い、3つ

誰が作るのか

誰でも作れる

サービスを出している側が、公式のものを作ることが多い

口を直接呼ぶのと何が違う

決め書きが、もうある

直接呼ぶなら、その決め方を自分で書くことになる

「道具を使う」と同じでは

別のこと。補い合う

MCP は誰が作るか、道具を使うは Claude がどう呼ぶか

肝心な違いは、誰がその仕事をするか

すでに誰かが作ってくれている

誰が作るのか 誰でも作れます。
サービスを出す側が公式のものを作
ることも多い

口を直接呼ぶのと何が違う MCP
なら、決め書きがもうあります

「道具を使う」と同じでは 別のこ
とです。補い合います

④ 肝心な違い



原石工房

よくある問い、3つ

誰が作るのか

誰でも作れる

サービスを出している側が、公式のものを作ることが多い

口を直接呼ぶのと何が違う

決め書きが、もうある

直接呼ぶなら、その決め方を自分で書くことになる

「道具を使う」と同じでは

別のこと。補い合う

MCP は誰が作るか、道具を使うは Claude がどう呼ぶか

肝心な違いは、誰がその仕事をするか

すでに誰かが作ってくれている

MCP サーバーは、道具の決め方と中身を、もう決めた形で渡します

「道具を使う」は、Claude が実際にその道具をどう呼ぶか、という話です

肝心な違いは、誰がその仕事をするか。MCP では、すでに誰かが作っています

⑤ 乗っかると、どうなるか



原石工房

MCP に乗っかると、どうなるか

自分で抱える

入り組んだつなぎこみの一式を、書いて・試して・手入れし続ける

MCP に乗っかる

外のサービスにつなぐ重い仕事を、サーバーが引き受ける

この連載では、第11回で一度 MCP に触れています

こちらは「外の道具につなぐ」という使う側の話でした

ここからの9回は、つなぐ側とサーバーを、自分で作る側へ回ります

※ この1行は原文ではなく、連載の案内です

入り組んだつなぎこみの一式を、自分で抱え続けなくてよくなります

外のサービスにつなぐ重い仕事を、サーバーが引き受けます

この連載では第11回でMCPに触れました。ここからは、作る側へ回ります

重い仕事を、引き受けてもらう

今回のまとめ



原石工房

MCP は、道具の決め方と動かす仕事を、専用のサーバーへ移す取り決め

MCP サーバーは、外のサービスへの、決まった形の窓口

誰でも作れる。サービスを出す側が公式のものを出すことも多い

「道具を使う」とは別のこと。違いは、誰がその仕事をするか

重い仕事を、引き受けてもらえる



Introduction to MCP の第2回

原題 MCP clients

訳 つなぐ側

自分のところとサーバーのあいだの橋が、どう働くのかを見ます

全51回の、33回目です

その道具、自分で作る必要がありますか



原石工房

隠れた才能に気づき、花開く