



# 第37回 「Introduction to Model Context Protocol」 Anthropic 公式 Claude Academy

## Claude Code

# 資料を定義する

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 /  
Defining resources

ここから、サーバーが差し出す2つめ「資料」に入ります

# 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37回

# 今日のレッスン

---



原石工房

## Introduction to MCP の第6回

原題 Defining resources

訳 資料を定義する

何かをさせるのではなく、情報を取ってきたいときに使います

# ① 例で見る



原石工房

## 原題 Understanding Resources Through an Example

何かをさせるのではなく、取ってくるためのもの

資料は、持っているデータを差し出すためのもの  
ウェブの「取ってくる側の受け口」と似ています

@書類の名前 で指せる仕掛けを作るなら

使える書類の名前を、全部もらう  
打っている途中に候補を出すため

指された書類の中身を、取ってくる  
実際に指されたとき

書類が指されると、その中身が  
Claude へ送る頼みごとの中へ、ひとりでに差し込まれます  
だから、Claude が道具を呼んで取りに行く必要がなくなります

@ で指すと、中身が入ってくる

@書類の名前 と打つと、その書類を  
指せる仕掛けを作るとします

要るのは2つ。名前を全部もらうこ  
とと、中身を取ってくること

指されると、中身が頼みごとの中  
へ、ひとりでに差し込まれます

## ② どう動くのか

### 原題 How Resources Work

資料は「頼んで、返る」形で動く

自分のプログラム

つなぐ側に、資料を頼む

つなぐ側

ReadResourceRequest を送る。住所 (URI) を添えて

MCP サーバー

住所を読み、合う働きを動かす

返ってくる

ReadResourceResult に、データが入っている

住所 (URI) が、どの資料がほしいかを指します

資料は「頼んで、返る」形で動きます

つなぐ側が ReadResourceRequest  
を送ります。住所 (URI) を添えて

サーバーが住所を読み、合う働きを動  
かし、結果を返します

道具を呼ぶのとは、別の道

### ③ 資料には2つの種類がある



## 原題 Types of Resources

資料には、2つの種類がある

#### そのままの資料

住所が決まっていて、変わらない

`docs://documents` 渡すものが要らない用事に

#### 差し込み口のある資料

住所の中に、渡すものが入っている

`docs://documents/{doc_id}` 道具立てが自動で読み取る

返す値は、道具立てが自動で形に直してくれます

自分で JSON の文字へ直す必要はありません

`mime_type` で、どんな種類のデータかを知らせます

住所に、渡すものを埋められる

そのままの資料 住所が決まっていて、変わらない

差し込み口のある資料 住所の中に、渡すものが入っている

道具立てが、そこを自動で読み取って働きへ渡します

## ④ 種類を知らせる



原石工房

### 原題 Implementation Details

mime\_type で、種類を知らせる

"application/json"

形のあるデータに

"text/plain"

ただの文字に

"application/pdf"

PDF のような中身に

検査の画面で試す

```
uv run mcp dev mcp_server.py
```

Resources

そのままの資料が並ぶ

Resource Templates

差し込み口のある資料が並ぶ

mime\_type で、どんな種類のデータかを知らせます

application/json は形のあるデータ、text/plain はただの文字

返す値は、道具立てが自動で形に直してくれます

つなぐ側が受け取る形が、そのまま見える



## 道具

呼ぶと、何かをしてくれる

Claude が「使おう」と決める

呼び出しが1回はさまる

書き換えもできる

## 資料

呼ばずに、中身を渡せる

@ で指すと、頼みごとに入る

道具の呼び出しが要らない

読むだけ

取ってくるだけなら、資料のほうが速い

## ⑤ 試す

---



原石工房

### 原題 Testing Your Resources

`uv run mcp dev mcp_server.py` で立ち上げます

Resources に、そのままの資料。Resource Templates に、差し込み口のある資料

つなぐ側が受け取ることになる答えの形が、そのまま見えます

# 今回のまとめ

---



原石工房

資料は、読むだけのデータを差し出すもの

@ で指すと、中身が頼みごとに入る。道具を呼ばなくてよくなる

種類は2つ。住所が決まっているものと、差し込み口のあるもの

mime\_type で、データの種類を知らせる

返す値の形直しは、道具立てがやってくれる

# 次回

---



原石工房

## Introduction to MCP の第7回

原題 Accessing resources

訳 資料を取りに行く

つなぐ側から、その資料を実際に取りってきます

全51回の、38回目です

それは、させたいことですか。知りたいことですか



原石工房

隠れた才能に気づき、花開く