



ClaudeCode×Obsidian活用研修 1日目

講師紹介

安田 光喜 (生成AIエバンジェリスト)



プロフィール／経歴

生成AI活用普及（技術検証、研究開発、研修設計、研修講師、受託開発）を中心に活動中。
札幌大学において人工知能やデータサイエンスも指導しながら、AI社員と協働する次世代の働き方を実証中。

【経歴】

公立はこだて未来大学 大学院（メディアデザイン領域）を卒業し、街づくり会社での複合商業ビルの立ち上げにおける情報インフラ整備やデザイン全般業務の責任として関与。その後デザイン事務所を立ち上げ、飲食店を中心としたデザイン業務や美術館展示のアートコンテンツ開発などを行う。また、地域ブランディングにも注力し、市主催の企画運営業務を行なった。2018年10月『小中学生向けのプログラミング教育』に注目し、函館市で初めての小中学生向けプログラミングスクール『D-SCHOOL北海道』を立ち上げて運営。その後北海道のドラッグストア『サッポロドラッグストアー』からスクールの買収を受け、教育を専門とするグループ会社『株式会社シーラクス』にて技術開発責任者として運営。スクール設計・運営、20以上の自治体連携（イベントや出張教室運営）、大人向けプログラミングスクールメンターなどさまざまな『次世代IT教育』に取り組む。

Dify公式資料の日本語翻訳や、SecondMeアンバサダーなど生成AI最新トレンドに注力。

#Cursor <AI駆動開発・AI駆動PM・スベック駆動開発> #Antigravity #GithubCopilot #Dify #n8n #KamuiOS

資格／認定

- ・修士（システム情報科学研究）
- ・生成AIパスポート / 基本情報技術者 / ITパスポート
- ・Adobe Illustrator認定エキスパート / G検定

得意分野／多言語経験

- ・生成AIをはじめとした世界最先端トレンド
- ・Python
- ・アプリケーション開発/自動化プログラム環境構築
- ・デザイン（DTP、UI、UX）
- ・教育（大学講師、研修、スクール運営など）

講演／研修実績など

- ・教育機関(小中高大)でのITトレンド講演多数
- ・自治体と連携した地域教育実績多数
- ・上場企業社員対象の生成AI研修多数

その他

私はこれまで何か1つの技術に特化するのではなく、ものすごくスピードの早い時代の流れにあわせ、さまざまな興味に対してマルチに取り組んでまいりました。ですので、1つの視点ではなく高い視座から広い視野での、“今後につながる”指導を心がけています。わかりやすく かつ 親身な指導を行ってまいりますので、ぜひお気軽にお声掛けください。

研修のスケジュール

ClaudeCode×Obsidian活用研修（1日目）

TIME:6h

● 座学主体 ● ハンズオン主体

■Session01:オリエンテーション [30min] 研修全体ゴール共有 | 2日間の流れ説明 | 受講スタンス確認 | 事前セットアップの最終確認 | アイスブレイク・質疑応答

■Session02:GenAI 2026年の地図 [60min]

- 各社LLMの位置付け OpenAIの特性 | Anthropicの特性 | Googleの特性 | オープンソース系の現在地 | モデル選定の軸 | その他LLM比較
- リサーチにおけるLLM リサーチエージェントの仕組み | ハルシネーション傾向 | 出典の扱い | 推論時の慎重さ | 実例比較 | リサーチのためのSkills実践
- Claude製品の使い分け Claude Code / Claude Desktop / Claude Design の位置付け | 業務シーン別の選び方 | コンサル業務への適合

■Session03:Obsidian活用ベストプラクティス [100min]

- インストールと初期設定 Vault作成 | テーマ選択 | 日本語フォント設定 | Markdown記法の基本 | フォルダ構造とコンテキスト | 画面構成と使い方
- フォルダ設計と運用思想 PARAメソッド | MOC(地図ノート) | Daily Noteの使い方 | 第二の脳としての運用 | タグ設計
- 必須プラグインとショートカット Templater / Dataview / Omnisearch / Advanced Tablesなどの紹介と導入 | 推奨ショートカット10選 | WebClip拡張

■Session04:Claude Code × リサーチmd化ハンズオン [150min]

- Claude Code基礎 インストールと認証 | VSCode連携 | 基本コマンド習得 | 権限モデルの理解 | 最初の対話を全員で体験
- リサーチ × md化ハンズオン 公開情報テーマ3案から選択 | 出典付きリサーチ指示 | Vault保存とタグ付け | ハルシネーション対策 | プロンプトテンプレート集活用

■Day1まとめ・中間課題説明・質疑応答 [20min]

*上記スケジュールは変更となる場合がございます。

事前セットアップ:Mac初期設定 | Obsidian / Claude Code / VSCode インストール

研修のスケジュール

ClaudeCode×Obsidian活用研修（2日目）

TIME:6h

● 座学主体 ● ハンズオン主体

■Session01:中間課題振り返りとDay2概要 [30min] 1週間の中間課題振り返り | 共通の詰まりポイント解消 | 良かった工夫の共有 | Day2到達点の明示 | 本日の制作物イメージ

■Session02:コンテキスト設計とエージェント活用 [85min]

- CLAUDE.md でAIを育てる プロジェクト前提の記述 | 書き方のコツ | 更新運用 | 効果の違いを実演で体感 | 自分用CLAUDE.md作成
- サブエージェントとSkill 役割特化の意義 | 独立コンテキスト | 並列実行の活用 | Skillによる定型化 | リサーチャー/エディター使い分け
- MCPサーバーによる拡張 MCPの位置付け | Obsidian連携 | Web検索連携 | ファイル操作連携 | コンサル業務でのユースケース

■Session03:ストーリーライン × スライドプラン構築 [130min]

- ストーリーライン設計 Why/What/How/So what 4層の型 | Claude Codeへの構造化指示 | MOCを用いた構成管理 | 内部リンク活用 | 自己検証プロンプト
- スライドプラン作成 スライドプラン作成デモ | スライド構造の渡し方 | AI駆動スライド生成の原則 | プラン出力テンプレート | md/JSON構造化 | ビジュアル要素の指定
- 骨子レビューと講評 聞き手目線の自己検証 | 2-3名の成果を全体で講評 | 講師フィードバック | 改善ポイント共有 | 業務転用の示唆

■Session04:DESIGN.md × python-pptx によるスライド化 [85min]

- DESIGN.md 設計と構造化 デザイン仕様書としての役割 | 配色/タイポ/余白の記述 | ADLテンプレ解析の正解例比較 | 受講者自身で構造化演習 | AI出力の安定化
- python-pptx 生成と発展デモ Pythonコード生成指示 | 全ページ自動生成デモ | 確認と微修正 | Claude Design発展デモ | python経路との使い分け

■クロージング・研修後ロードマップ・質疑応答 [30min]

*上記スケジュールは変更となる場合がございます。

事前セットアップ:中間課題の提出状況確認 | Python環境セットアップ状況 | 貴社テンプレート共有状況確認

この講座のゴール

ClaudeCodeとObsidianを連携し
人間×AI共通の『第2の脳』を設計する土台を作る
＋ClaudeCodeでリサーチ→資料作成を高速化する

詳細なゴールとゴールでないこと

最低限クリアすること

- ・ ClaudeCodeの使い方を習得する
- ・ コンテキスト、ハーネスの概念を理解する
- ・ Obsidianに知識が蓄積する仕組みを理解し、メンテナンスできる
- ・ 生成AIを活用したリサーチ→提案資料作成を行う
- ・ ClaudeCodeに相談しながら自走して疑問の解消やトラブル解決ができる

できればクリアすること

- ・ ClaudeCodeのハーネスを1から設計できる
- ・ Vaultにパーソナライズの情報を組み込む

目指していないこと

- ・ ClaudeCodeのすべてのハーネス設計を体験、理解する
- ・ AIエージェント開発の能力を向上する
- ・ Obsidianを使いこなせるようになる
- ・ 2日で第2の脳が完成する
- ・ トラブルやエラーが一切なく最後まで進行する

本研修への参加方法

事前配布を手元で確認しながら

『正しい知識の習得に特化した』研修です

- ＊『聞く』→『手元で調べる』姿勢をもち、正しく理解することを心がけましょう
- 本日で基本を抑え、今後さらにAIを活用していくための前提知識を持ちましょう

質問がある方： 直接質問



挙手 

可能な限り対応いたします

研修のスケジュール

ClaudeCode×Obsidian活用研修（1日目）

TIME:6h

● 座学主体 ● ハンズオン主体

■Session01:オリエンテーション [30min] 研修全体ゴール共有 | 2日間の流れ説明 | 受講スタンス確認 | 事前セットアップの最終確認 | アイスブレイク・質疑応答

■Session02:GenAI 2026年の地図 [60min]

- 各社LLMの位置付け OpenAIの特性 | Anthropicの特性 | Googleの特性 | オープンソース系の現在地 | モデル選定の軸 | その他LLM比較
- リサーチにおけるLLM リサーチエージェントの仕組み | ハルシネーション傾向 | 出典の扱い | 推論時の慎重さ | 実例比較 | リサーチのためのSkills実践
- Claude製品の使い分け Claude Code / Claude Desktop / Claude Design の位置付け | 業務シーン別の選び方 | コンサル業務への適合

■Session03:Obsidian活用ベストプラクティス [100min]

- インストールと初期設定 Vault作成 | テーマ選択 | 日本語フォント設定 | Markdown記法の基本 | フォルダ構造とコンテキスト | 画面構成と使い方
- フォルダ設計と運用思想 PARAメソッド | MOC(地図ノート) | Daily Noteの使い方 | 第二の脳としての運用 | タグ設計
- 必須プラグインとショートカット Templater / Dataview / Omnisearch / Advanced Tablesなどの紹介と導入 | 推奨ショートカット10選 | WebClip拡張

■Session04:Claude Code × リサーチmd化ハンズオン [150min]

- Claude Code基礎 インストールと認証 | VSCode連携 | 基本コマンド習得 | 権限モデルの理解 | 最初の対話を全員で体験
- リサーチ × md化ハンズオン 公開情報テーマ3案から選択 | 出典付きリサーチ指示 | Vault保存とタグ付け | ハルシネーション対策 | プロンプトテンプレート集活用

■Day1まとめ・中間課題説明・質疑応答 [20min]

*上記スケジュールは変更となる場合がございます。

事前セットアップ:Mac初期設定 | Obsidian / Claude Code / VSCode インストール

問い

『AGIに到達したか？』

AtCoder World Tour Finals 2026でAIが人間に完全勝利(2026.07.09)

#	名前	得点	時間	A (900点)	B (900点)	C (1500点)	D (2500点)	E (2500点)
	OpenAI AI	8300点	6:30:36	30:05	32:11	28:48	2:54:24 (誤答 2)	5:55:36 (誤答 5)
1	jiangly 人類	1800点	1:51:07	27:21	1:51:07			
2	turmax 人類	1800点	2:27:57	1:42:16 (誤答 1)	2:22:57			
3	gblaasnicse 人類	1800点	2:40:48	2:40:48	42:16			
4	tourist 人類	1800点	3:00:01	2:50:01 (誤答 2)	46:45			
5	Kevin090228 人類	1800点	3:52:45	1:10:43	3:47:45 (誤答 1)		(誤答 1)	
6	liuhengxi 人類	1800点	4:15:08	3:55:08 (誤答 4)	3:43:39			
7	tour1st 人類	900点	0:58:13		58:13		(誤答 2)	
8	ksun48 人類	900点	1:18:48	1:08:48 (誤答 2)				
9	noimi 人類	900点	1:59:31	1:59:31				
10	Nachia 人類	900点	2:26:41	(誤答 2)	2:26:41			
11	peti1234 人類	0点	0:00:00	(誤答 1)				
11	ecnerwala 人類	0点	0:00:00					

※正解した問題の誤答 1 件あたり、ペナルティとして 5 分が加算されます

AtCoder World Tour Finals 2026でAIが人間に完全勝利(2026.07.09)

三

YouTube JP

検索

Q

マイク

+ 作成

ベル

光る

WORLD TOUR FINALS 2026

Algorithm Japanese

Elapsed 03:47:54

Remaining 03:12:05

Sponsored by

SMBC OpenAI

BROADCAST DESK

CONTEST VENUE

CONFERENCE ROOM

liuhengxi

```
for(int i=0; i<10; i++) {
    at_vector<int> v(783);
    note: candidate: 'constexpr std::vector<Tp, Allocator>::vector(const_iter_type) [with Tp = std::pair<long long int, int>; Allocator = std::allocator<std::pair<long long int, int>>]
    783
    operator(initializer_list_value_type...)
    783
    for(int i=0; i<10; i++) {
        at_vector<int> v(783);
        note: no known conversion for argument 1 from 'std::remove_reference_t<vector<std::pair<long long int, int>; std::allocator<std::pair<long long int, int>>>' to 'std::initializer_list<std::pair<long long int, int>>'
        783
        operator(initializer_list_value_type...)
        783
    }
    shell returned 1
    Press ENTER or type command to continue
    bf.cpp:50:12: error: no match for call to '(dfs(int, ll)::lambda(int, ll, std::vector<std::pair<long long int, int>; std::allocator<std::pair<long long int, int>>>)'
    (int& long long int&, std::vector<std::pair<long long int, int>> &v)
    50 |         ch[ch[u][1]].first[ch[u][1].second];
      |         ~~~~~^~~~~~
    bf.cpp:49:18: note: candidate: 'dfs(int, ll)::lambda(int, ll, std::vector<std::pair<long long int, int>; std::allocator<std::pair<long long int, int>>>)'
    49 |         auto ch=&[&](int v, ll w, vector<pair<ll, int>> &v) {
      |         ~~~~~^~~~~~
    bf.cpp:49:18: note: no known conversion for argument 3 from 'std::vector<std::pair<long long int, int>>' to 'std::vector<std::pair<long long int, int>>'
    bf.cpp:50:12: error: no match for call to '(dfs(int, ll)::lambda(int, ll, std::vector<std::pair<long long int, int>>>)'
    (int& long long int&, std::vector<std::pair<long long int, int>> &v)
    50 |         ch[ch[u][1]].first[ch[u][1].second];
      |         ~~~~~^~~~~~
    bf.cpp:49:18: note: candidate: 'dfs(int, ll)::lambda(int, ll, std::vector<std::pair<long long int, int>; std::allocator<std::pair<long long int, int>>>)'
    49 |         auto ch=&[&](int v, ll w, vector<pair<ll, int>> &v) {
      |         ~~~~~^~~~~~
    bf.cpp:49:18: note: no known conversion for argument 3 from 'std::vector<std::pair<long long int, int>>' to 'std::vector<std::pair<long long int, int>>'
    shell returned 1
    Press ENTER or type command to continue
```

liuhengxi Submissions

Time	Problem	Score	Result
03:45:13	A	0	WA
03:44:55	A	0	WA
03:43:39	B	0	AC
03:43:39	B	0	AC
03:43:39	B	0	AC

/ STANDINGS /

R	User	Score	Time	A	B	C	D	E
1	liuhengxi	1000	03:45:13	0	0	0	0	0
2	tanukineiri	1000	02:27:57	0	0	0	0	0
3	gkkskksk	1000	02:40:48	0	0	0	0	0
4	tanuki	1000	02:00:01	0	0	0	0	0
5	tanuki	1000	02:00:01	0	0	0	0	0
6	tanuki	1000	02:00:01	0	0	0	0	0
7	tanuki	1000	02:00:01	0	0	0	0	0
8	tanuki	1000	02:00:01	0	0	0	0	0
9	tanuki	1000	02:00:01	0	0	0	0	0
10	tanuki	1000	02:00:01	0	0	0	0	0
11	tanuki	1000	02:00:01	0	0	0	0	0

/ EXHIBITION STANDINGS /

R	User	Score	Time	A	B	C	D	E
1	liuhengxi	1000	03:45:13	0	0	0	0	0

/ SUBMISSIONS /

User	Time	Problem	Score	Result
liuhengxi	03:45:13	A	0	WA
liuhengxi	03:44:55	A	0	WA
liuhengxi	03:43:39	B	0	AC
liuhengxi	03:43:39	B	0	AC
liuhengxi	03:43:39	B	0	AC

AtCoder World Tour Finals 2026 Algorithm Japanese stream

AtCoder Live

チャンネル登録者数 2.26万人

メンバーになる

チャンネル登録

153

共有

...

1万回視聴 2 日前に配信済み #プログラミング #エンジニア #AtCoder

チャットのリプレイ

すべて

提供: AtCoder Live

関連動画

>

上位のチャット...

トップファン

...

X

@drken1215 なんも分からない.....やばい

@tanukineiri もう半分なのか

@terry_u16 驚き屋になりました

@TorinikuSubs E問題良い問題

@cocoamilky1261 最新モデル使って3時間半ってAWTF...ガチ最難関なんだね

@--zu2oc DとEを両方解く気であるecnerwalaさんの調子はどうだろうか

@drken1215 ABC 30分以内に全完するような人たちが.....

@cocoamilky1261 chokudaiさんがコードアシスタント only AIをAtcoderから出すとか?

@wogikaze_sub 人間 vs AIのチーム戦の機運?

@drken1215 maroonさんの作問 vs AIが、最終決戦になっている。

@cocoamilky1261 Atcoder最高の瞬間

@revogen-r2v ゴクッ

@terry_u16 アルゴサイト中にご飯食べるのむずそう

@dsu-j6o 打つのはっや

@--zu2oc 128-bit 整数を使おうとしてる

Copyright © Givory, Inc. All rights reserved https://info.atcoder.jp/more/contents/awtf/2026?_gl=1*19hw7it*_ga*MTY2MjMjgNDEyMC4xNzc0QDgyNDM0*_ga_RC512FD18N*czE3ODM4MTI2MzlkZkYkZkZkdE3ODM4MTI2MzlkajYwJGwwJGJGww

12

ChatGPT、Gemini東大理Ⅲに主席合格(2026.LifePrompt様調べ)

東大 首席比較表

 LifePrompt

科類	配点	ChatGPT 5.2 Thinking	Claude 4.5 Opus	Gemini 3 Pro Preview
理科一類	550	503.59	451.99	496.54
理科二類	550	503.59	451.99	496.54
理科三類	550	503.59	451.99	496.54
文科一類	550	452.70	426.18	460.07
文科二類	550	452.70	426.18	460.07
文科三類	550	452.70	426.18	460.07

※ オレンジの数値は、各科類の合格者最高点を超過していることを示します。

※ 配点は、各科類の合計点（満点）を示します。

【各科類の合格者最高点（参考）】

・ 理科一類：443.28

・ 理科二類：396.85

・ 理科三類：453.60

・ 文科一類：430.13

・ 文科二類：420.85

・ 文科三類：434.96

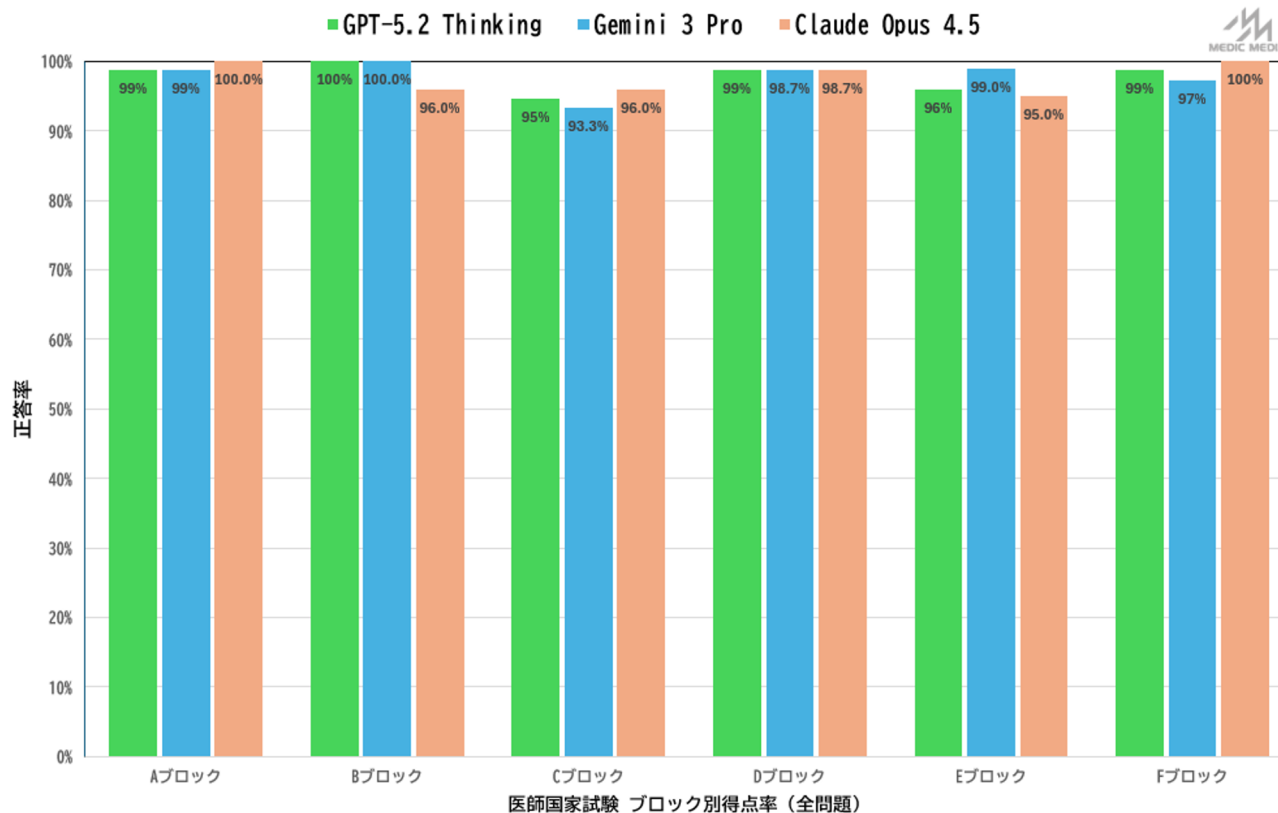
※ 上記データは東京大学が公表した2026年度一般選抜（前期日程）における合格者の実績に基づきます。

医師国家試験であらゆるAIが95%以上(2026)



	必修		一般	
	得点	割合	得点	割合
GPT-5.2 Thinking	196点/200点	98.0%	292点/300点	97.3%
Gemini 3 Pro	199点/200点	99.5%	290点/300点	96.7%
Claude Opus 4.5	191点/200点	95.5%	295点/300点	98.3%

医師国家試験であらゆるAIが95%以上(2026)



2025-2026で加速した生成AI分野

・ Agentic AI（AIエージェント）の実用化（2025-2026の最大トレンド）

単に質問に答えるだけでなく、目標を与えて複数ステップのタスクを自律的に実行するAI。ITサポート、研究、顧客対応、ワークフロー自動化などで広く導入。Gartnerが2025年のトップ戦略技術に挙げ、企業での採用が急増。企業全体戦略として本格化し、ROI（投資収益率）向上や業務効率化が実証され始めた。

・ 生成動画（Generative Video）の成熟

テキストや画像から高品質動画を生成する技術（Sora、Veo、Klingなどの進化版）が実用レベルに。映画・広告・教育・エンタメで活用爆増。リアルタイム性や音声同期も向上し、クリエイティブ業界の生産性が劇的に変わった。

・ ヘルスケア・ライフサイエンス

薬剤発見の加速（分子構造提案・シミュレーション）、画像診断の精度向上、パーソナライズド治療、仮想細胞モデルなど。AIが新薬開発タイムラインを大幅短縮。医療機器の智能化も進む。

・ 製造業・ロボティクス（特にヒューマノイドロボット）

AI統合ヒューマノイドロボット（Tesla Optimus、Boston Dynamics Atlas、Figureなど）が工場実装やフィールドテスト段階に。労働力不足対策として注目。家事支援やサービス用途への展開も始まる。

『AGIに到達したか？』

本日の環境

Claude CodeとObsidianの連携全体図



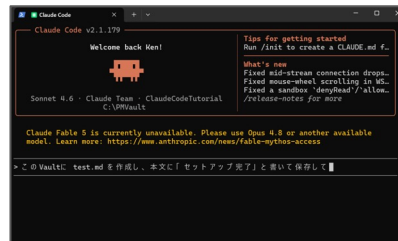
Claude : 人間に代わって操作するエージェント

ユーザー設定

ユーザーディレクトリ/~~.claude

ユーザー設定（危険なコマンドの実行禁止や、確認画面の省略など）はClaudeCodeが動作する全ての環境において適用されます

ユーザー設定のフォルダはClaudeCodeインストール時に指定したパスに自動で作成されます



プロジェクト設定

配布フォルダ/~~.claude

研修などで実際に用いるプロジェクトフォルダです
ユーザー設定よりこちらの設定が優先されます



claudeが書き込み



読み込み

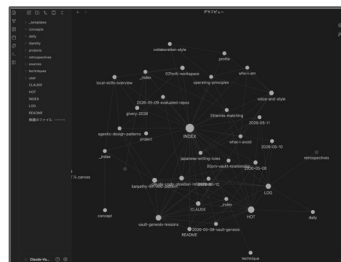


Obsidian : 本研修における自分専用のAIのデータベース

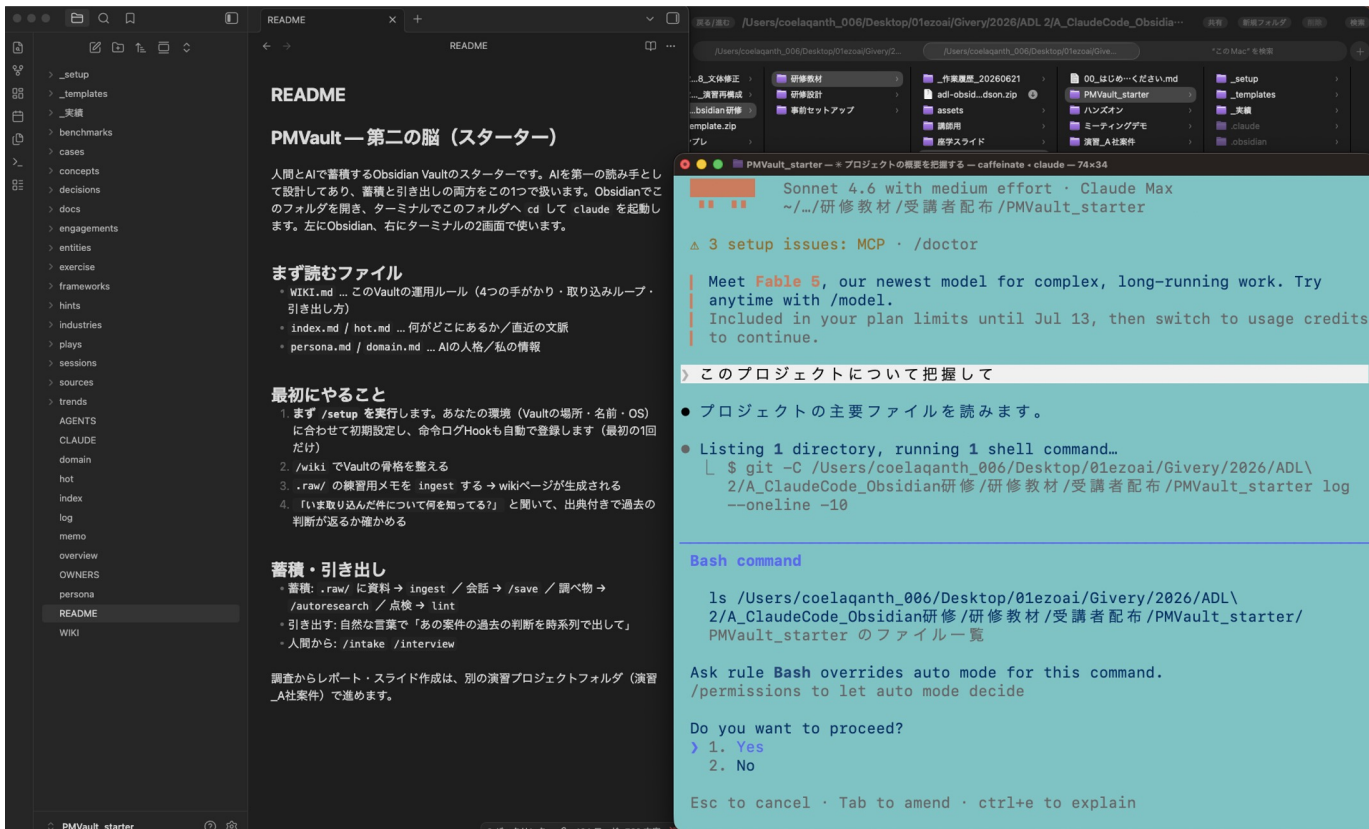
PM-Vault（ローカル保管庫）

本研修用の任意の場所に空の保管庫を作成ください

Claudeのユーザー設定と別に蓄積していく業務のナレッジ
や人間の知見を収集・蓄積・整理します

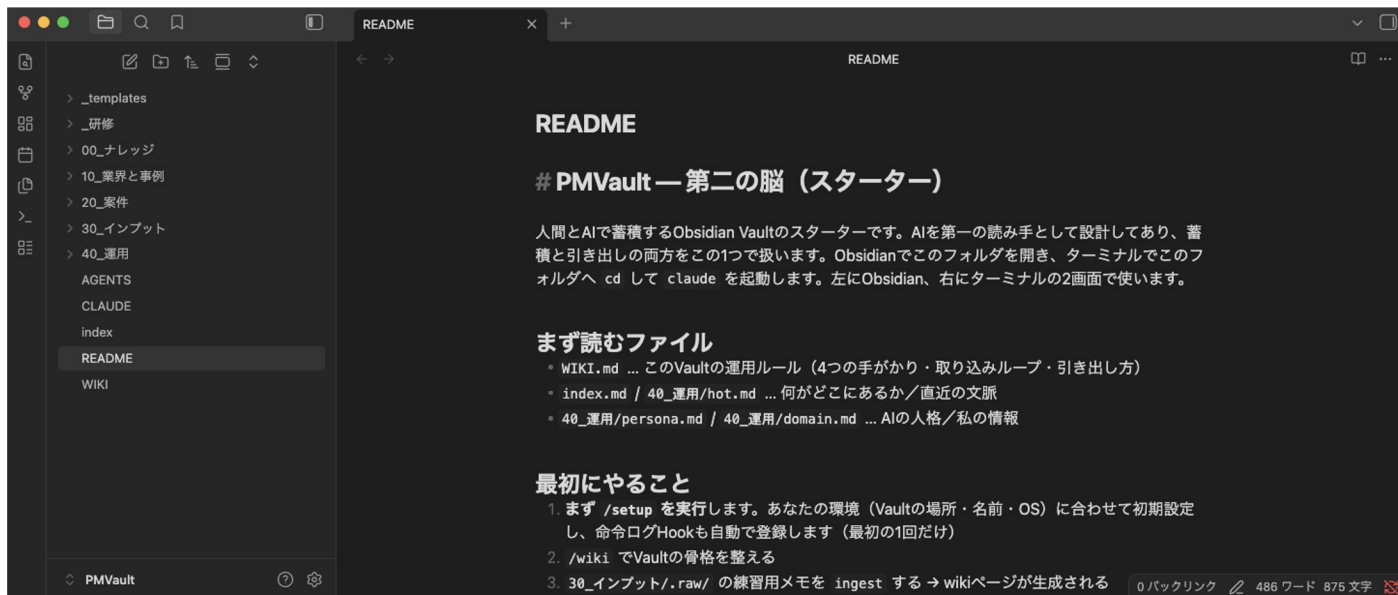


Obsidian × Claude (AgentView)



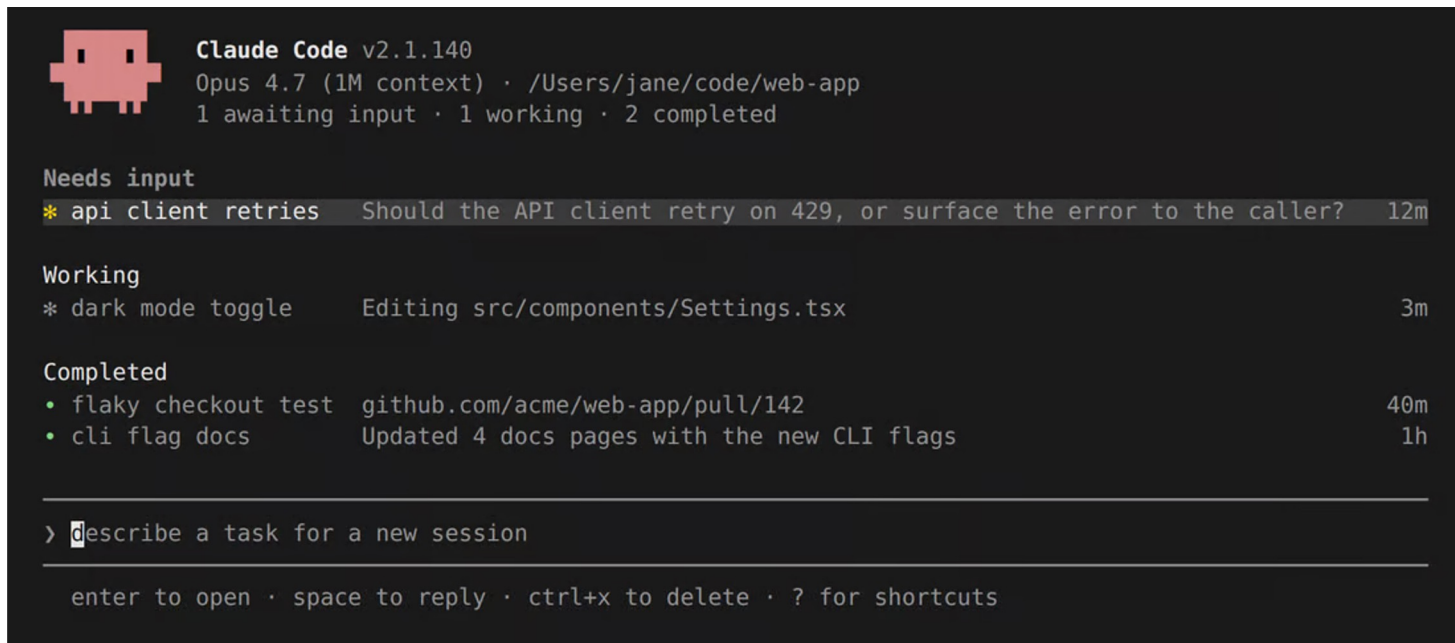
Obsidian側の設定

- ①配布フォルダを展開する
- ②Documentsフォルダ内に『PMVault』フォルダを移動
- ③Obsidianを起動し、保管庫を開く から Documents/PMVault を選んで開く
- ④開いた内容が画像と同じ構造を確認し、目を通す



ClaudeCode側の設定

- ① ターミナル/コマンドプロンプトなどを開く
- ② cdコマンドで Documents/PMVault に移動
- ③ claude agents コマンドでAgentViewを起動



Claude Code v2.1.140
Opus 4.7 (1M context) · /Users/jane/code/web-app
1 awaiting input · 1 working · 2 completed

Needs input

* api client retries	Should the API client retry on 429, or surface the error to the caller?	12m
----------------------	---	-----

Working

* dark mode toggle	Editing src/components/Settings.tsx	3m
--------------------	-------------------------------------	----

Completed

• flaky checkout test	github.com/acme/web-app/pull/142	40m
• cli flag docs	Updated 4 docs pages with the new CLI flags	1h

>  describe a task for a new session

enter to open · space to reply · ctrl+x to delete · ? for shortcuts

Tips : AgentViewとは

Claudeを別々のプロジェクトごとに並列で起動し、PCを閉じてでも会話を維持できる
並列・継続的にClaudeCodeを活用するための公式機能

基本的に十字キーで移動する

Pinned : タブ固定

→Ctrl + T

Ready for review : AI動作完了

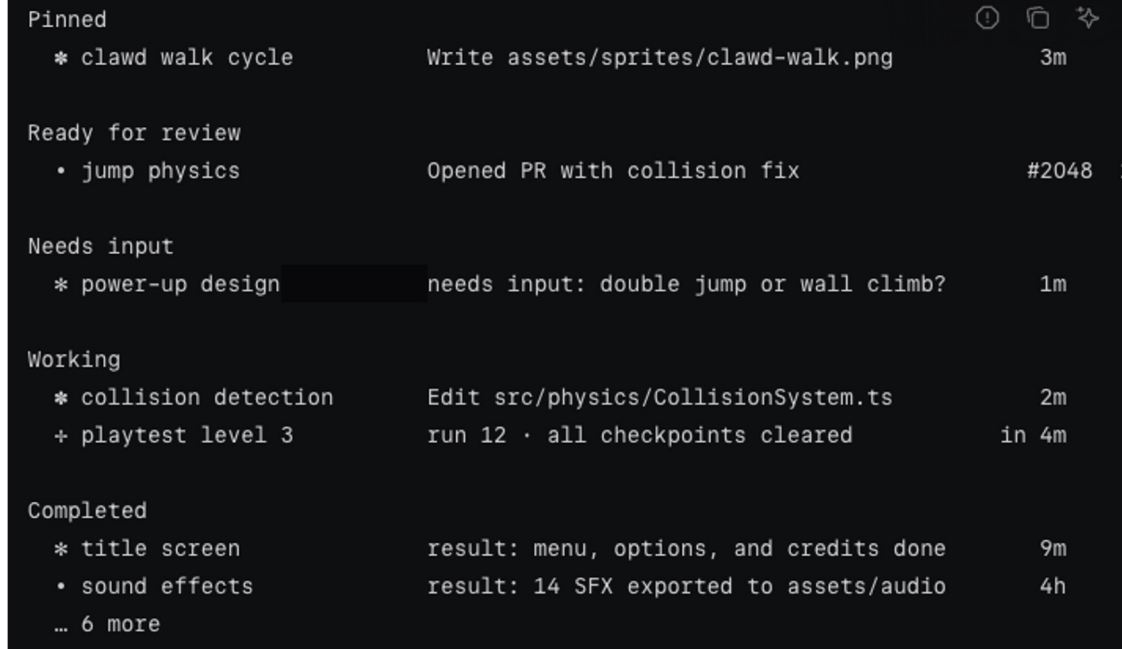
→人間の確認待ち

Needs input : 人間の承認待ち

Working : 稼働中

Completed : タスクの達成

...6 more : Enterで表示



Pinned			ⓘ	📄	🌟
* clawd walk cycle	Write assets/sprites/clawd-walk.png	3m			
Ready for review					
• jump physics	Opened PR with collision fix	#2048	2		
Needs input					
* power-up design	needs input: double jump or wall climb?	1m			
Working					
* collision detection	Edit src/physics/CollisionSystem.ts	2m			
+ playtest level 3	run 12 · all checkpoints cleared	in 4m			
Completed					
* title screen	result: menu, options, and credits done	9m			
• sound effects	result: 14 SFX exported to assets/audio	4h			
... 6 more					

生成AI 2026年の最新地図

ClaudeCode活用デモ

リサーチ→スライドテンプレートに合わせてPPTX資料作成

テーマを募集します

デモの解説

- ①デモのためのフォルダを作成します
- ②そのフォルダをClaudeCodeの作業場所として指定します
→調査データの蓄積、PPTXなどの保存のために必須
- ③調査→PPTX作成まで自動で行うコマンド + 調査テーマの指示プロンプト を送信
- ④ClaudeCodeが調査→データ蓄積→データ取りまとめ→テンプレート読み込み→骨子作成
→PPTX作成→セルフチェック→修正→報告 まで自動で動作
- ⑤調査レポートが完成します

調査→PPTX作成まで自動で行うコマンド = /search2pptx コマンドとして
ユーザー設定（PC内のClaudeCodeであればどこでも呼び出せる設定）として作成

作成時のプロンプト >>

調査→PPTX作成まで自動で行うコマンドを生成したプロンプト

ユーザー設定（いつでもどの場所でも呼び出せる設定）で、/search2pptx <レポート作成テーマ> というコマンドを打てば、ClaudeCodeが与えられたテーマについて

①調査テーマを把握 ②目的や用途、スライドを用いる相手、枚数、説明の粒度（文字多め、図多め、画像多めなど）など不明瞭な部分を人間側にUserAskInputで質問 ③人間の回答を踏まえて大規模検索 ④調査先

のサイト、資料、論文の一覧を作成し、カテゴリや優先度をつける ⑤ここまでの作業内容をmdにまとめ、可視化したHTML資料をawesome-design-skillから1つランダムにDESIGN.mdを選んで作成。ブラウザを自動で

開いてHTML資料を表示し、人間に『これらの情報元と優先度に対してイメージのズレがないか』確認を取る。 ⑥人間がOKを出したら次へ。NGや修正指示があればmd、HTMLを修正して再度確認。 ⑦次に内容を要約

、集約、整理し、作成しようとしているスライドのページ数、構成、見出しとその中身を並べたスライドドラフトをmdで作成。画像(スクショなど)や図、表はassetsフォルダに保存したり、.pyでその部分の図を

作成できるようなスクリプトとして書き出し、その解説をmdに入れ、できたら人間に確認を取る。その際スクショや図、表、引用などの情報も詳細に記述する。 ⑧そのスライドドラフトで問題なければスライド

を生成するスクリプトを作成→セルフチェック ⑨実際にPPTXを生成、生成されたPPTXのページすべてをスクショで保存してimgaesフォルダに格納。その画像を自身で再度セルフチェックし、レイアウト崩れやわか

りにくい部分があればページの増減も踏まえて修正。 10.修正を繰り返し、最終的にチェック用の複数の視点でチェックするSubagents（Opusなどハイモデル）がすべてOKを出せば完成とし、動作完了。人間に通

知する。

上記のようなパイプラインを、人間のスラッシュコマンド起点で起動できるようにSkills、Subagents、Hooks、MCP、Commandsを最適に設計してください。

各操作においては最適なモデルを選び、トークンの管理を心がけること

また、この命令はultracodeで起動すること。

調査→PPTX作成まで自動で行うコマンド

作成されたハーネスと設定

- **Command /search2pptx <テーマ>**

＝ 10ステップの司令塔。作業ディレクトリ~/search2pptx/<slug>/に成果物を貯めながら進行

- **Skills 6本**

＝ intake(要件)/research(検索・キュレーション)/review-viz(HTML可視化)/slide-draft(構成)/build(PPTX+スクショ)/review-panel(最終ゲート)

- **Subagents**

＝ web-researcher(並列検索・haiku)/slide-critic(多視点チェック・opus)

- **MCP**

＝ WebSearch/WebFetch(検索)、playwright(ブラウザ表示・スクショ)、PushNotification(完成通知)

- **Hook**

＝ 完成通知+PPTX生成後の自動スクショ

- **モデル配分**

＝ 検索haiku/統合・可視化・buildsonnet/検索戦略・スライド設計・criticはopus

Dyanmic Workflow機能での実装の様子

```
Subagents = WebSearch(WebFetch(検索)、playwright(ブラウザ表示・スクショ)、PushNotification(完成通知)
- MCP = WebSearch/WebFetch(検索)、playwright(ブラウザ表示・スクショ)、PushNotification(完成通知)
- Hook = 完成通知 + PPTX生成後の自動スクショ
- モデル配分 = 検索haiku/統合・可視化・buildsonnet/検索戦略・スライド設計・criticはopus

ultracodeで各コンポーネントを並列実装して、整合監査まで通す。settings.jsonへのHook追記だけは自己改変になるので、実装は「hookスクリプト作成+追記内容の明示」に留めて、有効化は最後に承認取る。

今から許可いただきたいのは Workflow で、search2pptxパイプライン式 (Command/Skills/Subagents/Hook) を並列実装し整合監査するという動作を期待するものです。

● Workflow(/search2pptx パイプライン (Command/Skills/Subagents/Hook) をユーザー設定に実装)
  └ Running in background · /workflows to monitor and save

● /search2pptx 一式の実装ワークフローが走り出した (Build 10コンポーネント→Verify 2監査→Fix)。設計の要点を置いてく。

作るもの (全部 ~/.claude/ 配下)
- commands/search2pptx.md ... 10ステップの司令塔
- skills/s2p-{intake,research,review-viz,slide-draft,build,review-panel}/SKILL.md ... 各段の専門手順
- agents/s2p-web-researcher.md (haiku・並列検索)、agents/s2p-slide-critic.md (opus・多視点チェック)
- hooks/search2pptx-notify.sh ... 完成通知
- ~/search2pptx/README.md ... 使い方 + settings.jsonへ足すHook設定のスニペット

設計のキモ

> 続けて
```

```
● How is Claude doing this session? (optional)
1: Bad 2: Fine 3: Good 0: Dismiss
```

```
build-search2pptx
/search2pptx パイプライン (Command/Skills/Subagents/Hook) をユーザー設定に実装 0/10 agents · 3m59s
```

Phases	Build · 10 agents	
> 1 Build 0/10	command (retry 1)	Opus 4.8 (1M context) 87.6k tok · 7 tools
2 Verify	intake	Sonnet 4.6 32.2k tok · 11 tools
3 Fix	research	Opus 4.8 (1M context) 39.3k tok · 3 tools · idle 3m 42s
	reviewviz	Sonnet 4.6 28k tok · 4 tools · idle 3m 44s
	draft (retry 1)	Opus 4.8 (1M context) 81.2k tok · 7 tools
	build (retry 1)	Opus 4.8 (1M context) 76.4k tok · 9 tools
	panel	Opus 4.8 (1M context) 39.3k tok · 4 tools · idle 3m 45s
	agent_researcher	Sonnet 4.6 28.4k tok · 6 tools · idle 3m 42s
	agent_critic	sonnet queued
	hook_readme	sonnet queued

各社LLMの位置付け

生成AIの最先端を牽引している存在

ANTHROPIC

Research

Economic Futures

Commitments ▾

Learn ▾

News

Try Claude

▾

AI research and products that put safety at the frontier

AI will have a vast impact on the world.
Anthropic is a public benefit corporation
dedicated to securing its benefits and
mitigating its risks.

FEATURED

FOUR HUNDRED
METERS *on* MARS

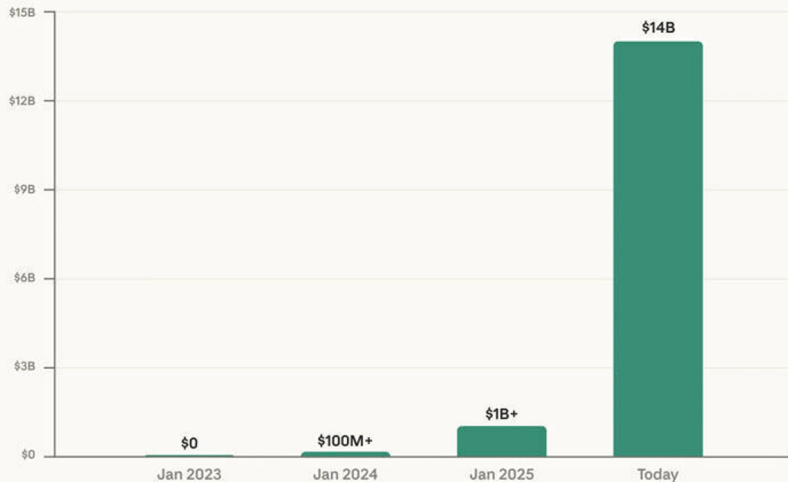
Anthropicの企業価値と推移

年月	資金調達ラウンド	企業価値 (時価総額)	特記事項・主な投資家
2023年2月	シリーズC頃	約37億ドル	Google等が参画
2023年5月	-	約40億ドル	-
2024年1月	-	約176億ドル	Amazon/Google出資
2025年3月	シリーズE	約580億～615億ドル	-
2025年9月	シリーズF	約1,830億ドル	130億ドルを調達
2026年2月	シリーズG	約3,800億ドル (約6.2兆円)	300億ドル調達、Microsoft/Nvidia参画
2026年5月	シリーズH	約9,650億ドル (15.6兆円)	約650億ドル調達、

2025年3月の約600億ドルから、1年足らずで約15.7倍の9,650億ドルへと急騰

Anthropicの企業価値と推移

Anthropic run-rate revenue growth



10X+

growth each of the past
three years

\$14B

run-rate
revenue today

売り上げ: 140億ドル規模

→過去3年間、毎年10倍以上のペース
で拡大

Claudeを年間10万ドル超利用する顧客
数は過去1年で7倍に増加。

さらに、年間100万ドル超を支出する
顧客は500社超え

Fortune 10企業のうち8社がすでに
Claudeを導入

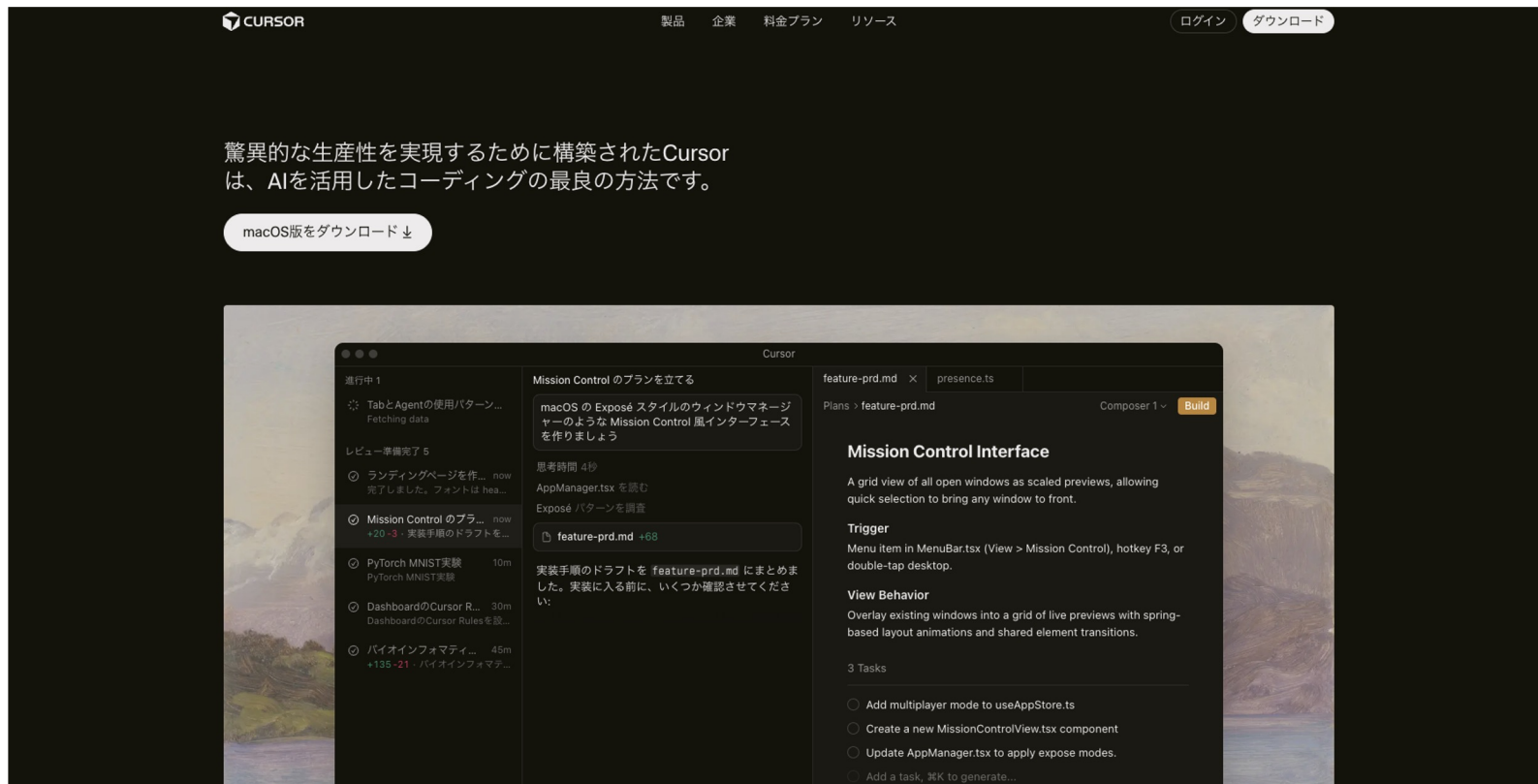
AI駆動開発環境の全体図：IDE

企業の環境ごとに最適なツール導入を検討。

Cursor、Antigravityは非エンジニアの方も利用率急増中

項目	Cursor	Antigravity	Kiro	VSCode+Copilot
ベース	VSCode	独自	VSCode	VSCode
月額	\$20～	未発表	AWS課金	\$10～
AIモデル	複数選択可	独自	Bedrock	OpenAI + Claude
拡張機能	VSCode互換	独自	VSCode互換	VSCode標準
企業導入	増加中	少	AWS顧客中心	最多
強み	Composer	新パラダイム	セキュリティ	エコシステム

Cursor



Antigravity

 Google Antigravity

Product

Use Cases ▾

Pricing

Blog

Resources ▾

Download 

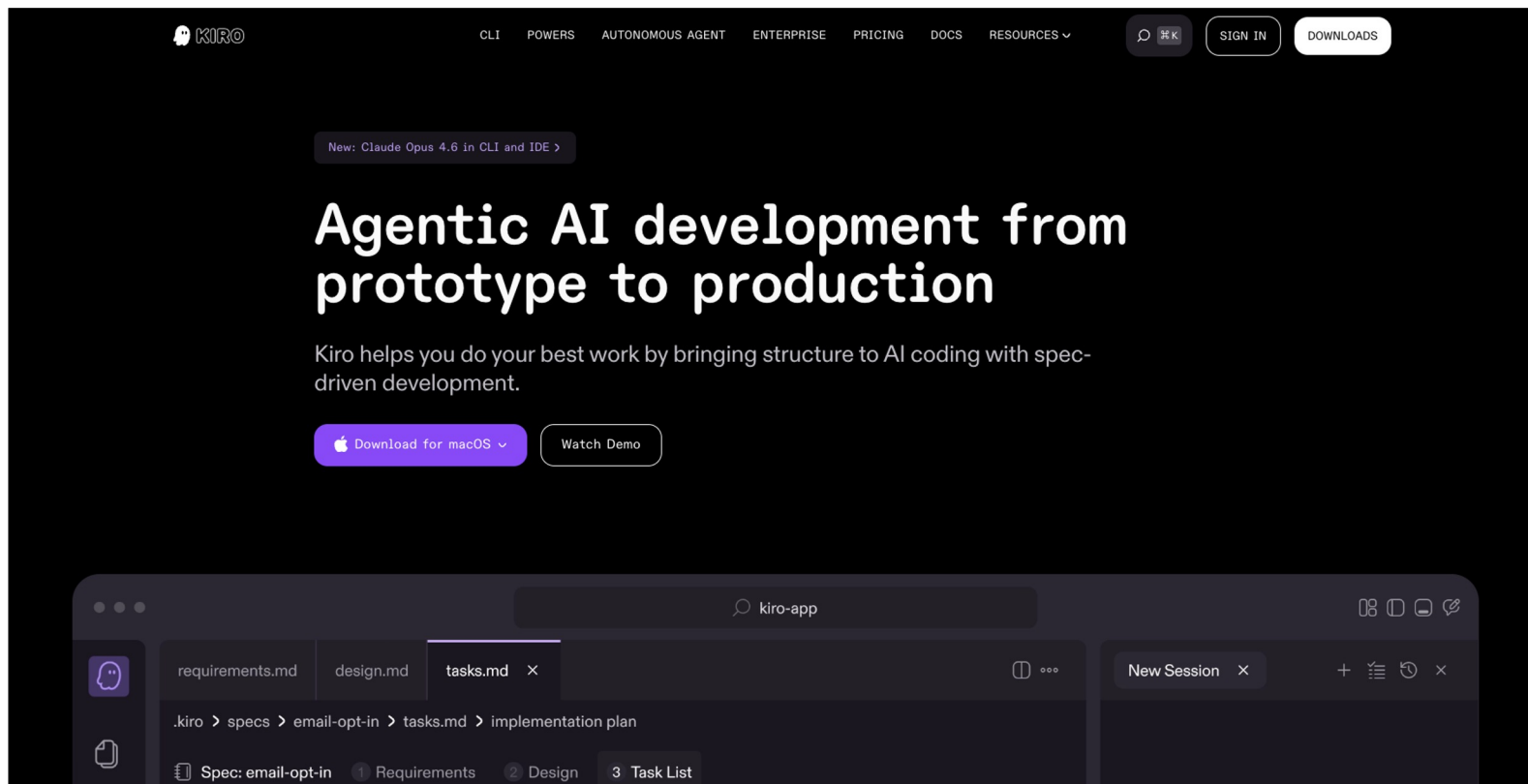
 Google Antigravity

Experience liftoff with the next-generation IDE

 Download for MacOS

Explore use cases

Kiro



VSCode × Github Copilot

The screenshot shows the Visual Studio Code documentation page for GitHub Copilot. The top navigation bar includes links for Visual Studio Code, Docs, Updates, Blog, API, Extensions, MCP, FAQ, and Events. A search bar and a 'Download' button are also present. The main content area is titled 'GitHub Copilot in VS Code' and features a sidebar with a table of contents. The 'Overview' section is currently selected, displaying text about multi-agent development capabilities. A right-hand sidebar titled 'ON THIS PAGE' lists links to various sections. Below this, there are two callout boxes: 'Get started with AI' and 'Plan a feature with agents'. An inset image shows a web browser displaying a '404 Not Found' error on the 'Wanderlust Blog' website, with a sidebar menu visible.

Visual Studio Code Docs Updates Blog API Extensions MCP FAQ Events

VS Code February events – Agent Sessions Day on Feb 19th

GitHub Copilot in VS Code

GitHub Copilot adds multi-agent development capabilities to Visual Studio Code. Plan your approach, then let AI agents implement and verify code changes across your project. Run multiple agent sessions in parallel: locally, in the background, or in the cloud. Manage them all from a central view. Inline suggestions, inline chat, and smart actions assist you throughout the rest of the coding workflow.

Agents and agent sessions

Agents handle complete coding tasks end-to-end. Give an agent a high-level task and it breaks the work into steps, edits files, runs terminal commands, invokes tools, and self-corrects when it hits errors or failing tests. Each task runs inside an **agent session**, a persistent conversation you can track, pause, resume, or hand off to another agent.

ON THIS PAGE

- Agents and agent sessions**
- What can you do
- Getting started
- More ways to code with AI
- Customize AI for your workflow
- Support
- Pricing
- Next steps

TRY THIS

Get started with AI

Follow a hands-on tutorial to build your first app with AI in VS Code.

[Start tutorial](#)

Plan a feature with agents

Use the Plan agent to create a structured implementation plan for a new feature.

[Open in VS Code](#)

File Edit Extensions View Go ... GitHub Copilot

Wanderlust Blog: Travel Adventures & Inspiration

Wanderlust Blog Home About Destinations Blog Contact

404

Not Found

Wanderlust Blog

Sharing amazing travel experiences and inspiring your next adventure around the world.

Quick Links

- Home
- About
- Destinations
- Blog
- Contact

Contact

Local Background Cloud Codes

全てに共通の『Rules』

どのAI駆動IDEであっても『プロジェクトにおけるルール』などに関しては名前は違ってもほぼ同様の書き方、考え方を適用できる

ツール	Rulesファイル	配置場所
Cursor	.cursorrules	プロジェクトルート
GitHub Copilot	copilot-instructions.md	.github/
Claude Code	CLAUDE.md	プロジェクトルート
Kiro	.kirorc	プロジェクトルート

CLIで動作するAI：ClaudeCode

Claude Code Docs 日本語

検索...

Ask AI Claude Developer Platform Claude Code on the Web

はじめに Claude Code で構築する デプロイ 管理 設定 リファレンス リソース

はじめに

Overview

クイックスタート

変更履歴

コア概念

Claude Code の仕組み

Claude Code を拡張する

一般的なワークフロー

ベストプラクティス

プラットフォームと統合

ウェブ上の Claude Code

Claude Code デスクトップ版

Chrome 拡張機能 (ベータ版)

VS Code で Claude Code を使用する

JetBrains IDEs

GitHub Actions

GitLab CI/CD

Slack での Claude Code

はじめに

Claude Code の概要

Claude Code は、コードベースを読み取り、ファイルを編集し、コマンドを実行し、開発ツールと統合する agentic coding ツールです。ターミナル、IDE、デスクトップアプリ、ブラウザで利用できます。

Claude Code は、機能の構築、バグの修正、開発タスクの自動化を支援する AI 搭載のコーディングアシスタントです。コードベース全体を理解し、複数のファイルとツール全体で作業して、タスクを完了できます。

はじめに

環境を選択してはじめましょう。ほとんどのサーフェスには、**Claude サブスクリプション**または**Anthropic Console** アカウントが必要です。Terminal CLI と VS Code は **サードパーティプロバイダー**もサポートしています。

Terminal VS Code Desktop app Web JetBrains

ターミナルで Claude Code を直接操作するための機能豊富な CLI です。ファイルを編集し、コマンドを実行し、コマンドラインからプロジェクト全体を管理できます。

To install Claude Code, use one of the following methods:

Native Install (Recommended) Homebrew WinGet

このページの内容

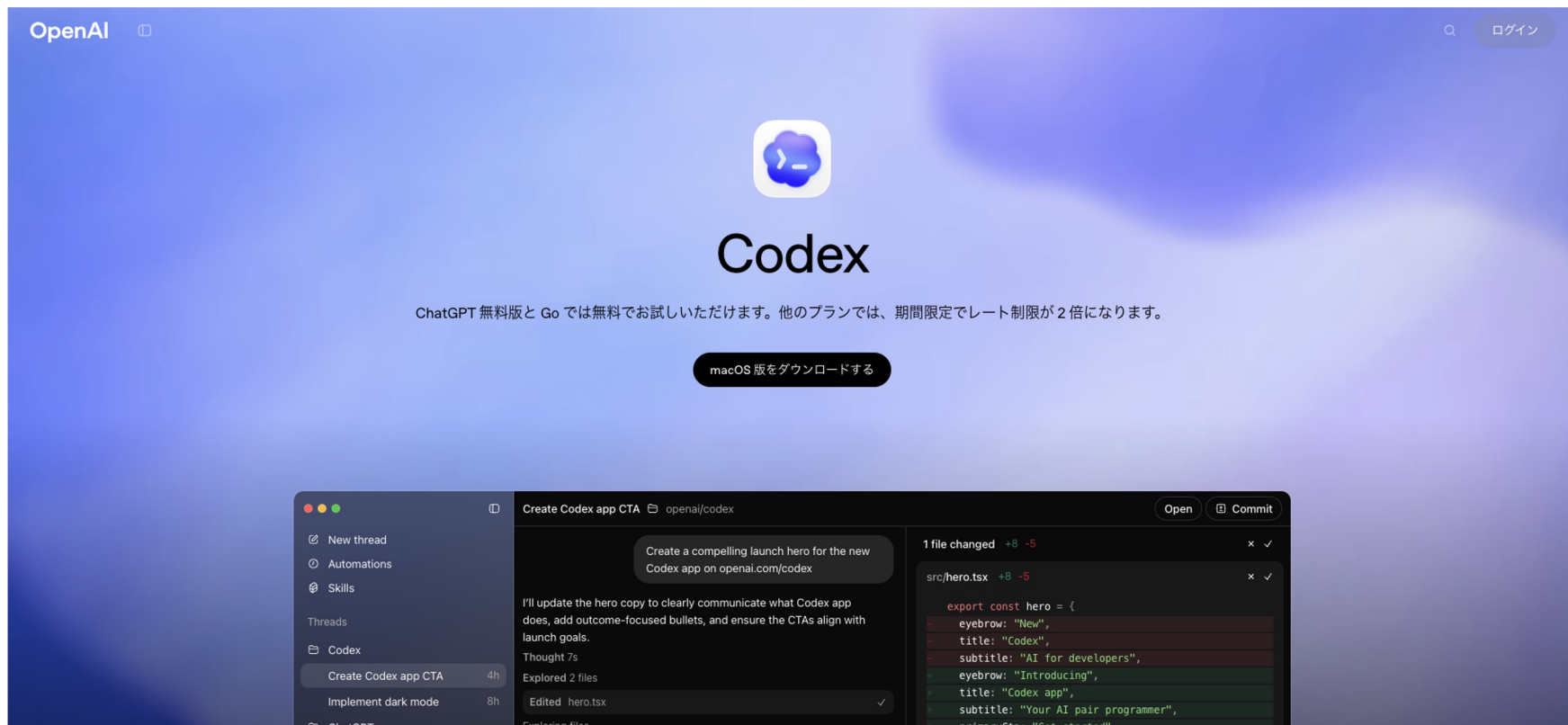
はじめに

できること

Claude Code をどこでも使用する

次のステップ

CLIで動作するAI：Codex



CLIで動作するAI： GeminiCLI

Google Cloud

お問い合わせ

無料で開始

ブログ

ソリューションとテクノロジー ▾

エコシステム ▾

デベロッパー

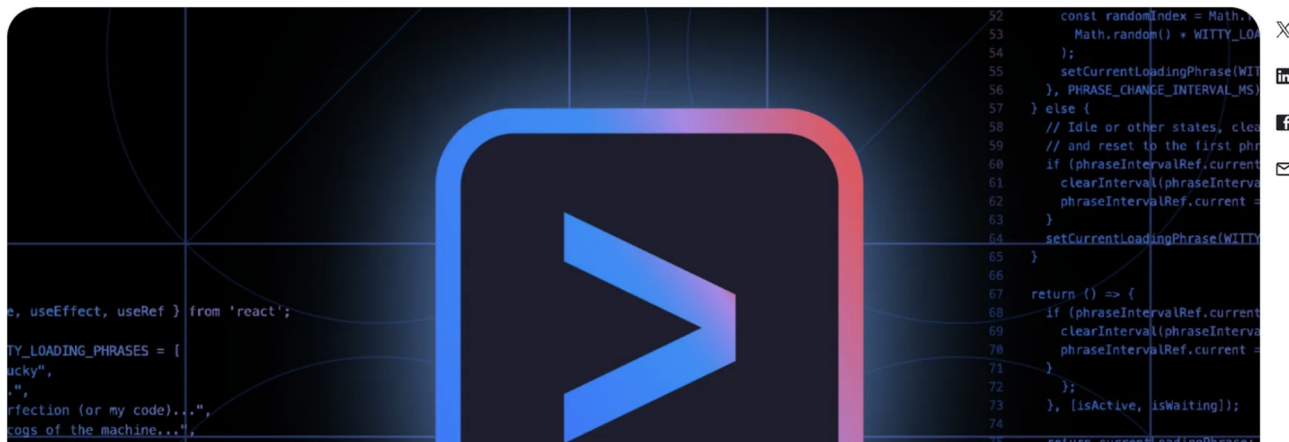
Transform with Google Cloud



デベロッパー

Gemini CLI : オープンソース AI エージェント

2025年6月26日



CLIで動作するAI：Codex

ターミナルやGhostty、iTermなどのCLI環境で動作するAIを用いた開発も増加

項目	Claude Code	Codex	Gemini CLI
提供元	Anthropic	OpenAI	Google
ベースモデル	Claude	GPT	Gemini
OSS	一部	あり	なし
エージェント機能	強い	中程度	中程度
クラウド統合	なし	Azure連携可	GCP強い
コンテキスト長	200K	128K	1M

その他ローカルLLM



LOCAL

Ollama

ローカルLLM実行の事実上の標準。1コマンドでモデルのダウンロードと起動が完了する。Llama、Codestral、DeepSeek Coderなど主要モデルに対応。REST APIを提供するためContinueやAvante等のエディタ拡張と連携可能。



LOCAL

LM Studio

GUIでモデルを管理・実行できるデスクトップアプリ。HuggingFace上のGGUFモデルをワンクリックでダウンロード。OpenAI互換APIを提供するので、既存のCopilot連携設定をそのまま流用できる。



EXTENSION

Continue

VSCode/JetBrains用の拡張機能。ローカルLLMをコーディングアシスタントとして接続する。Ollamaとの連携が容易で、Copilotと同等のタブ補完・チャット・Agent機能をローカルモデルで実現。

リサーチにおけるLLM

リサーチエージェントの仕組み

- **Layer 1: LLM** (Reasoning Engine) ← Claude Opus/Sonnetなど
- **Layer 2: Reasoning Loop / Orchestration** (LangGraph, CrewAIなど)
- **Layer 3: Tools** (Web Search, Browser, Code Execution, RAG)
- **Layer 4: Memory** (Short-term: Context / Long-term: Vector DB + Obsidian Vault)
- **Layer 5: Observability & Guardrails** (Logging, Evaluation, Human-in-the-Loop)

ClaudeにおけるWeb検索

ClaudeCodeのWeb検索は、AnthropicのサーバーサイドWeb Searchツールを基盤としています（[Claude.ai](#)のチャット版と同じ）。

主な特徴：自動判断で起動：ユーザーの自然言語指示（例：「React 18.3の新機能教えて」）に対して、Claudeが必要と判断した場合に自動で検索を実行。

主なバックエンド：**Brave Search**を主に活用

2つの主なWebツール：

- ①WebSearch：検索クエリから関連ページのタイトル・URLを取得（軽量）
- ②WebFetch：特定URLのページ内容を取得・要約（質問に対する回答形式で返す）。

ClaudeCodeはこれらを**Tool Use（ツール呼び出し）**で自律的に使い分けます。

*MCP（Model Context Protocol）経由で外部検索エンジン（Googleなど）を拡張接続することも可能

検索の動作原理

- ①**ユーザー入力の理解**：ターミナルなどの指示をClaudeモデル（Sonnetなど）が解析。
- ②**検索クエリの生成**：最適な検索クエリをAIが内部生成（必要に応じて複数）。
- ③**WebSearch実行**：Anthropicサーバーが検索を実行し、タイトル・URL・ページ年齢などの結果を返す
- ④**結果分析と判断**：取得したリンクから関連性を評価。必要ならWebFetchで特定ページの内容を取得。
- ⑤**WebFetchの処理**：URL検証 → ドメイン安全チェック → HTML取得（キャッシュ15分） → Markdown変換 → 小型モデル（Haikuなど）で質問に対する要約・回答生成。
- ⑥**追加検索（必要時）**：情報が不足していればループ。
- ⑦**回答生成**：検索結果 + モデル知識を統合。引用（出典リンク）付きで回答。
- ⑧**表示**：ターミナルに要約・コード例・ソースリンクを表示。

リサーチにおけるハルシネーション傾向

テーブル（モデル | 代表ベンチマークでのHallucination Rate | 傾向）。

- ・ Claude Opus/Sonnet系：低～中（特に拒否行動が強い。AA-Omniscienceで低hallucination）。
- ・ GPT-5.x系：中～高（創造性が高い分、自信過剰）。
- ・ Gemini系：検索統合で改善傾向（Deep Researchで低減）。

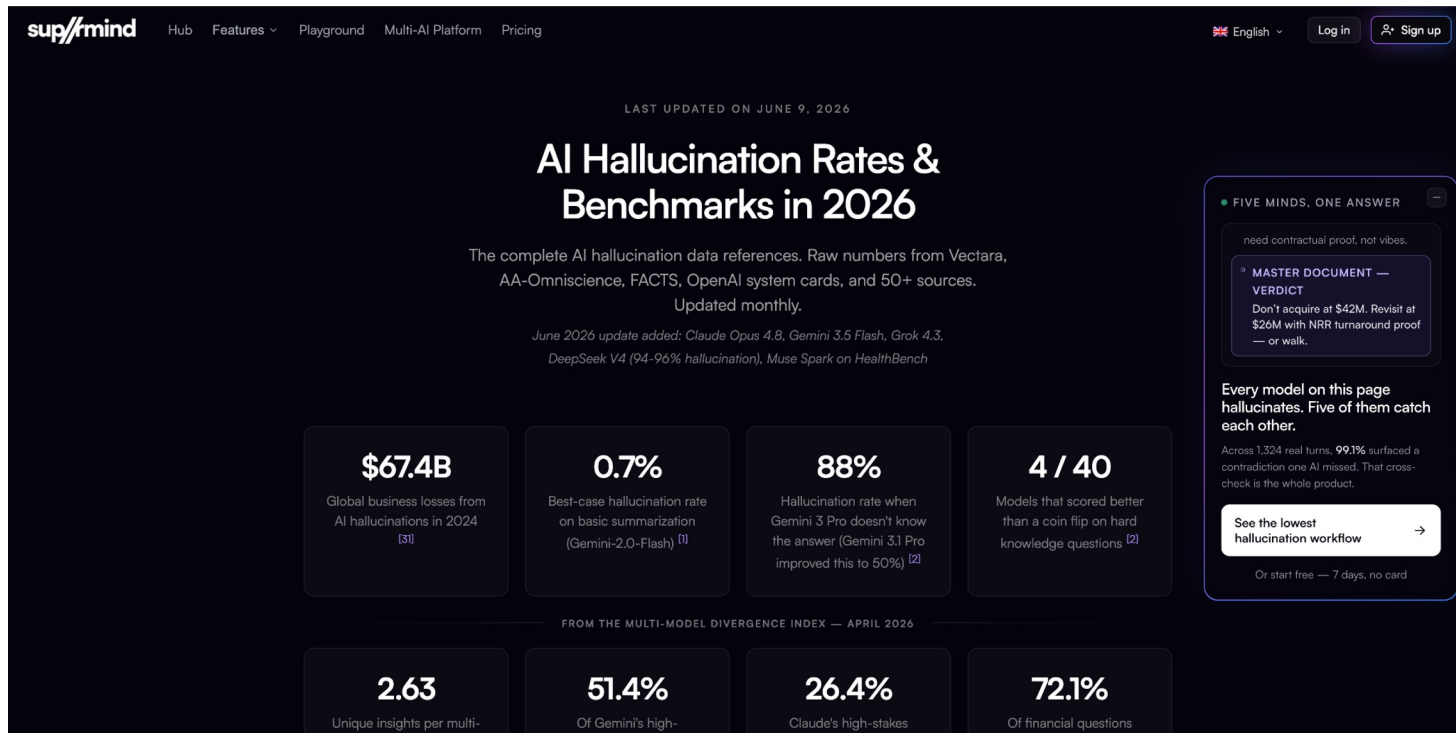
<https://suprmind.ai/hub/ai-hallucination-rates-and-benchmarks/>

原因・対策3原則:

- ①出典必須指示 + 未知は「不明」と明記。
- ②Chain-of-Verification (CoVe) プロンプト。
- ③複数モデル/検索でクロスチェック。

＊Claudeの強み: 憲法AIにより「不確実時は控える」傾向が強い → リサーチに適する。

suprmind.ai : ハルシネーションのスコア一覧サイト



出展の扱い

なぜ出典が重要か（コンサル業務視点）

→LLMは引用を「生成」するが、自動検証ではない。2026年でも最終責任は人間。

良い出典扱いの例

Markdownで [出典: URL](タイトル) + アクセス日 + 要約根拠。

プロンプト例:

「以下のテーマについて、信頼できる公開情報から調査してください。

各主張に具体的な出典URLを明記し、信頼度（High/Med/Low）を付与。ハルシネーションは厳禁。不明点は正直に述べること。」

Obsidian連携

→出典リンクを自動でWikiリンク化、Dataviewで出典リスト自動生成。

ハルシネーション対策L1：プロンプトで対策

随時以下のプロンプトを選んで含める

＊ハルシネーションを避けて回答してください（Apple公式）

＊回答の根拠には、必ずURLソースをつけてください

＊ステップバイステップで回答してください

＊与えた知識以外の情報は回答に含めないでください

＊もし情報がなければわからないと回答してください

L2：ツールや機能で対策する

- ①**検索機能**で最新情報を取得する手順を入れる
- ②AIが自身で事実確認できる手順を組み込む（**ファクトチェック**）
- ③**RAG**（Retrieval-Augmented Generation）を用いて知識を読み込めるようにする

L3：AIのベースから対策する

- ①**高品質なデータセット**を利用（不純物の多いデータセットを省く）
- ②教科学習を行ない、AIのモデルを**ファインチューニング**する
- ③AIの**自信・確信度**を調整する内部評価を組み込む

生成AIを使う際、必ず留意すること

①許可されているツール以外は使用しない

会社が指定または許可した、セキュリティが確保された生成AI環境のみを利用してください。**個人で契約しているサービスや、一般に公開されている無料のWebサービスを無断で業務に利用すると、情報漏洩に繋がるリスクがあります。**

②機密情報・個人情報をプロンプトに入力しない

生成AIへの指示（プロンプト）には、**会社が許可していない情報は入力しないでください。**許可されたツール、許可された情報の範囲を正しく理解し、生成AIを制約の中で利用しましょう。入力したデータがAIの学習に使われ、意図せず外部に流出する可能性があります。

③AIの生成物は必ずレビューし、無検証で利用しない

生成AIが作成したコード、文章、設定値などは、**必ず人間の目でその正確性と安全性をレビューしてください。**AIは平気で間違いや脆弱性を含む回答（ハルシネーション）を生成します。無検証での利用は、システム障害やセキュリティインシデントの直接的な原因となります。

※業務で使用する場合は、必ず就業先のルールを確認し、許可されている範囲内で利用する。

各モデル比較表

観点	Claude系	ChatGPT / OpenAI系	Gemini系
基本傾向	慎重・保守的・論理整合性重視	バランス型・発想力と実務処理が強い	検索・Google連携・情報収集に強い
推論の進め方	前提確認、条件整理、リスク指摘が得意	目的に合わせた回答生成、企画化、整形が得意	Web情報やGoogle Workspace文脈を使った調査が得意
不確実な情報への態度	不明点を明示しやすい傾向	もっともらしく補完することがある	検索結果に引っ張られることがある
ハルシネーションリスク	低めになりやすいがゼロではない	創造的補完で発生することがある	出典付きでも要確認
出典付き回答	Web検索機能がある場合、出典確認に向く	ブラウジング時は出典付きで強い	Deep Research系で強い
推論速度	深く考えるほど遅くなる	比較的速く、整形も得意	検索込みだと時間がかかる
長文・複雑タスク	文脈保持と構造化に強い	幅広いアウトプット形式に強い	大量情報の探索と統合に強い
反論・リスク検討	強い	指示すれば強い	情報収集後の比較に強い
研修での推奨用途	調査方針設計、論点整理、報告書骨子	提案書化、説明文、比較表、ワーク設計	最新情報収集、出典探し、Google資料連携
注意点	慎重すぎて答えが抽象化することがある	きれいにまとめすぎて根拠が薄くなることもある	検索結果の質に依存する

各推論(Effort)の傾向

レベル	状態	AIの出力の特徴	向いている場面	危険な場面
低い慎重さ Low	速いが粗い	すぐ答える、根拠が薄い、断定が多い	アイデア出し、ラフ案、たたき台	法務、医療、経営判断、出典必須の調査
中程度の慎重さ Midium	実務で使いやすい	前提をある程度整理し、一般的なリスクも触れる	研修資料、企画案、業務整理	最新情報・数値・制度変更が絡む調査
高い慎重さ High	検証重視	不明点、前提、反証、出典確認を含める	リサーチ、コンサル資料、意思決定支援	速度優先のプレスト
過剰な慎重さ xHigh	動きが遅い	判断を避ける、抽象論が増える、結論が弱い	高リスク領域の初期分析	短時間で方向性を出す場面

実例比較(天秤.AI byGMO)

ネットのセキュリティも GMO 無料診断

GMO 大会議

あかげさまで 30.3 連勝 GMO

天秤.AI byGMO

新しいチャット

チャットを検索

共有・保存済み履歴

天秤の使い方・AI最新情報

主要AI 約款比較

ユーティリティ

Deep Research

画像生成

プロンプトを呼び出す

外部AIリンク集

履歴

Difyの機能と特徴を視覚化...

Dify 1DAY研修プログラム計画

北海道の生成AI取り組みとデ...

「海辺での穏やかなひととき」

Free アップグレード

標準

高度・専門

使用済み: 0 / 80

user_1919263710227... Freeプラン

おはようございます、user_191926371022754816さん

無料お試しビジネスプラン

Gemini 3.5 Flash

GPT 5.4

Claude Sonnet 5

Perplexity Web検索

Grok 4.3

PLaMo 3.0

質問してみましょう。⌘+Enterで送信

+

検証プロンプト

あなたは慎重なリサーチコンサルタントです。

以下のテーマについて、公開情報をもとに調査し、コンサル資料の下書きとして使える形で整理してください。

テーマ：

2026年現在、日本企業が生成AIを業務導入する際に重視すべきポイント

目的：

生成AI導入を検討している企業の管理職向けに、意思決定の材料を作ること。

出力条件：

1. 最初に結論を5行以内で要約する
2. 次に、重要論点を5つに整理する
3. 各論点について「背景」「企業への影響」「注意点」を書く
4. 可能な限り出典を付ける
5. 出典が確認できない内容は「要確認」と明記する
6. 推測と事実を分けて書く
7. 最後に、企業が次取るべきアクションを3つ提案する

評価したい観点：

- ・回答速度
- ・論点整理のうまさ
- ・出典の妥当性
- ・ハルシネーションの少なさ
- ・不明点を不明と言えるか
- ・コンサル資料として使えるか

出力形式：

Markdown形式。

見出し、表、箇条書きを使って読みやすく整理する。

おはようございます、user_191926371022754816さん

無料お試しビジネスプラン



Gemini 3.5 Flash



GPT 5.4



Claude Sonnet 5



Perplexity Web検索



Grok 4.3



PLaMo 3.0

あなたは慎重なリサーチコンサルタントです。

以下のテーマについて、公開情報をもとに調査し、コンサル資料の下書きとして使える形で整理してください。

テーマ：

2026年現在、日本企業が生成AIを業務導入する際に重視すべきポイント

目的：

生成AI導入を検討している企業の管理職向けに、意思決定の材料を作ること。

出力条件：

+

↩

↑

検証プロンプト

The screenshot displays the Tenbin AI workspace interface. On the left is a sidebar menu with options like '新しいチャット' (New Chat), 'チャットを検索' (Search Chats), and '共有・保存済み履歴' (Shared/Saved History). The main area contains several open chat windows for different AI models:

- Gemini 3.5 Flash (消費:4)**: Discusses the challenges of AI adoption in Japanese companies, mentioning high costs and the need for change management.
- GPT 5.4 (消費:1)**: Focuses on '2. AIガバナンス・法規制・社内ルール整備' (AI Governance, Regulations, and Internal Rule Development), discussing the impact on companies and the need for licenses.
- Claude Sonnet 5 (消費:4)**: Provides a response to a prompt about being a heavy research consultant, discussing themes related to AI adoption in Japanese companies in 2026.
- Perplexity Web検索 (消費:1)**: Discusses the code optimization, modularization, and lightweighting of AI technologies, and the impact on companies.
- Grok 4.3 (消費:4)**: Discusses the implementation of AI usage guidelines by 2026, security, and the need for management training.
- PLaMo 3.0 (消費:4)**: Provides another response to the prompt about being a heavy research consultant, discussing themes related to AI adoption in Japanese companies in 2026.

At the bottom, there is a chat input field with a placeholder '質問してみましょう。⌨+Enterで送信' (Ask a question. Send with ⌨+Enter) and a '送信先 6 / 6' (Send to 6 / 6) indicator.

リサーチのためのSkills実践

Claude製品の違い

Claude.ai (ブラウザ)



Claude.ai（ブラウザ）

標準的なClaudeのWebインターフェース。ProjectsやArtifactsが使いやすい基本形。

主な用途：日常会話、文章作成、アイデア出し、調べ物

向いている人：初心者～一般ユーザー

苦手：ローカルファイル操作や大規模開発

国内活用事例

- ・マーケティング部門での市場調査・広告コピー作成（NTTデータ事例）
- ・法務部門での契約書レビューと要約（大手企業バックオフィス）

Claude Desktop（デスクトップアプリ）



Claude Desktop（デスクトップアプリ）

ブラウザ版を拡張したPC専用アプリ。Voice ModeやExtensionsに対応。

主な用途：日常業務全般、デスクトップ作業との連携

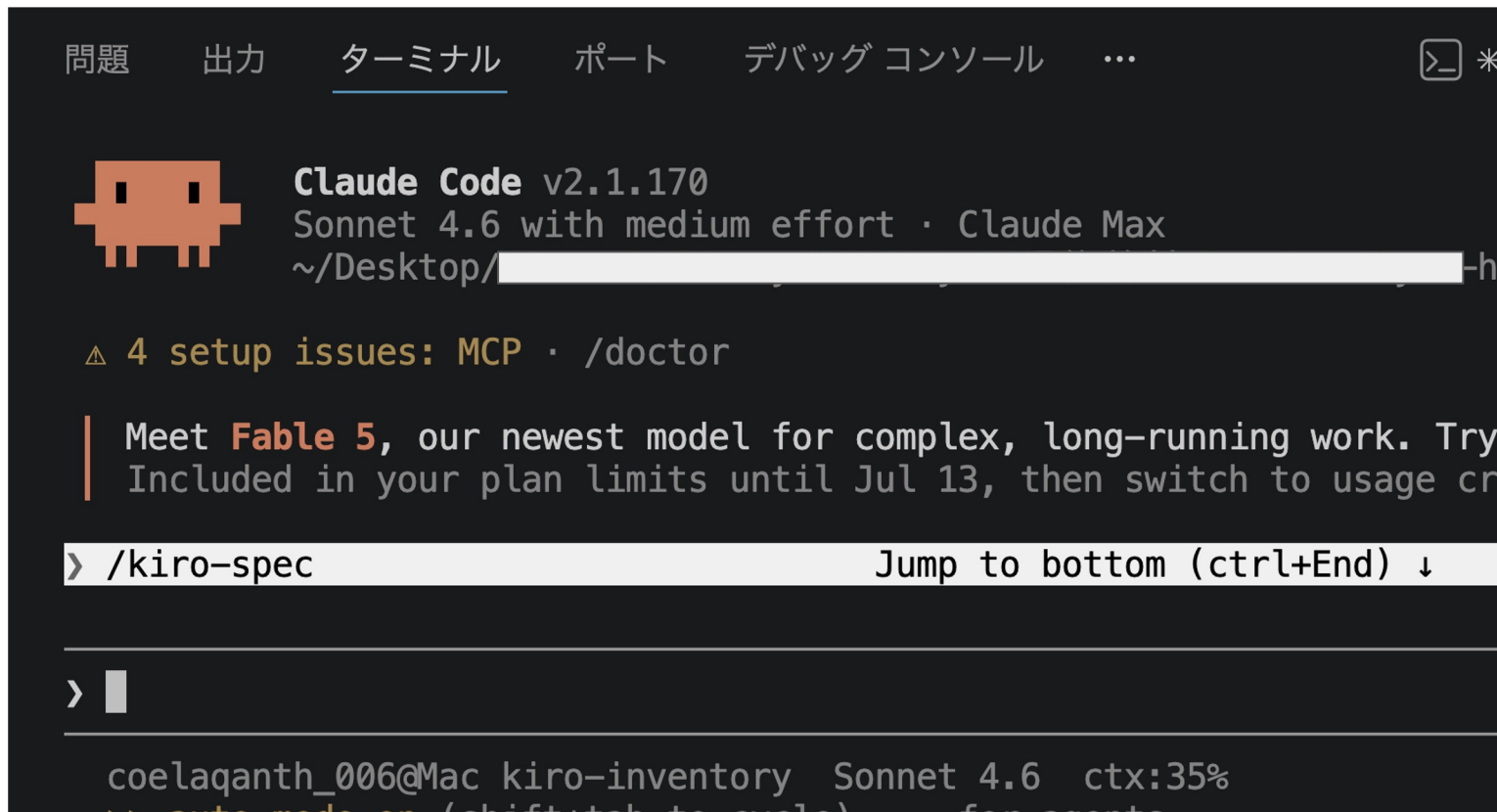
向いている人：ヘビーユーザー

特徴：Chat・Cowork・Codeを1つのアプリで切り替え可能

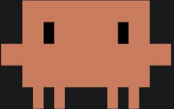
国内活用事例

- ・経費精算書自動作成（Rimo社PdM実践例）
- ・業務資料の自動整理とレポート生成（複数企業導入）

Claude Code (CLI版)



問題 出力 ターミナル ポート デバッグ コンソール ...

 **Claude Code** v2.1.170
Sonnet 4.6 with medium effort · Claude Max
~/.Desktop/

⚠ 4 setup issues: MCP · /doctor

Meet **Fable 5**, our newest model for complex, long-running work. Try
Included in your plan limits until Jul 13, then switch to usage cr

> /kiro-spec Jump to bottom (ctrl+End) ↓

> █

coelaqanth_006@Mac kiro-inventory Sonnet 4.6 ctx:35%

Claude Code (CLI版)

ターミナルベースの開発者向けエージェント。Gitやコマンド実行に強い。

主な用途：大規模コード開発、リファクタリング、自動化

向いている人：ソフトウェアエンジニア

苦手：非技術者にはハードルが高い

国内活用事例

- ・ バグ修正とテスト自動化 (IT企業エンジニア活用)
- ・ レガシーシステムのリファクタリング支援

Tips : AgentView



Claude Code v2.1.140

Opus 4.7 (1M context) · /Users/jane/code/web-app

1 awaiting input · 1 working · 2 completed

Needs input

* **api client retries** Should the API client retry on 429, or surface the error to the caller? 12m

Working

* **dark mode toggle** Editing src/components/Settings.tsx 3m

Completed

- **flaky checkout test** github.com/acme/web-app/pull/142 40m
- **cli flag docs** Updated 4 docs pages with the new CLI flags 1h

> **d**escribe a task for a new session

enter to open · space to reply · ctrl+x to delete · ? for shortcuts

Claude Code（ブラウザ・デスクトップアプリ版）



Claude Code（ブラウザ・デスクトップアプリ版）

GUIで使えるClaude Code。ターミナル不要。

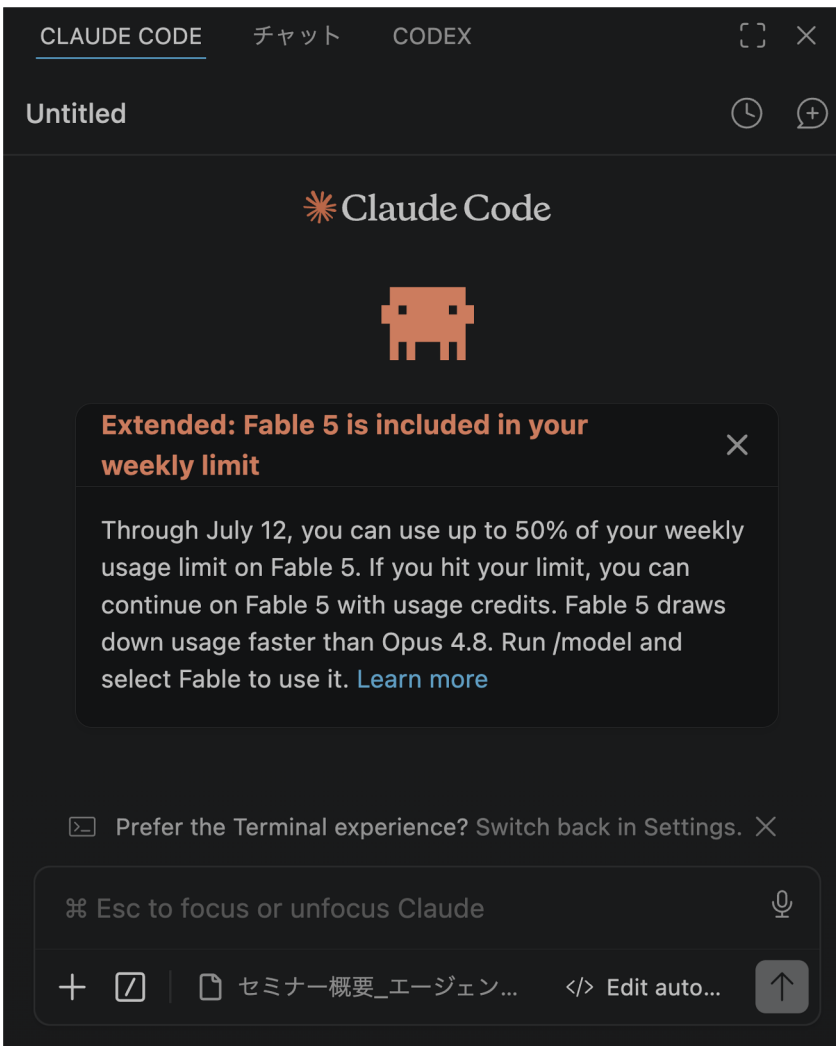
主な用途：コード作成・編集（視覚的に確認したい場合）

向いている人：開発者だがGUIを好む人

国内活用事例

- ・ Webサイトの企画から実装まで（ソウゾウ社サービス事例）
- ・ 業務アプリのプロトタイプ開発

Claude Code 拡張プラグイン



Claude Code拡張プラグイン

エディタ内で直接動作する拡張機能。

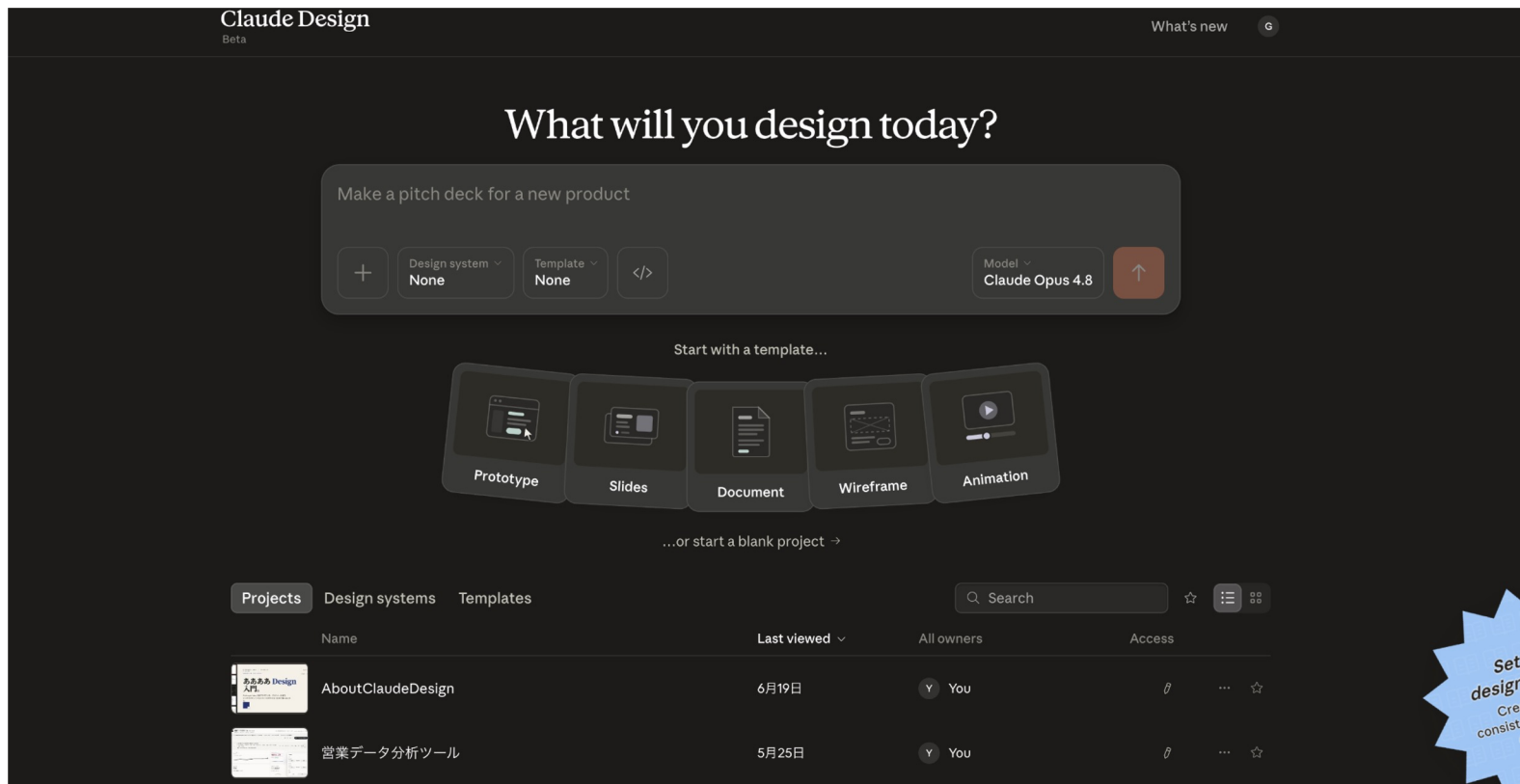
主な用途：リアルタイムコード補完・デバッグ

向いている人：普段VSCodeを使う開発者

国内活用事例

- ・日々のコーディング支援とエラー解析（エンジニアコミュニティ実践）
- ・API開発の高速化

Claude Design (ブラウザ・デスクトップアプリ)



Claude Design

ビジュアルデザイン専用ツール。プロトタイプやスライド作成に特化。

主な用途：UIデザイン、モックアップ、プレゼン資料

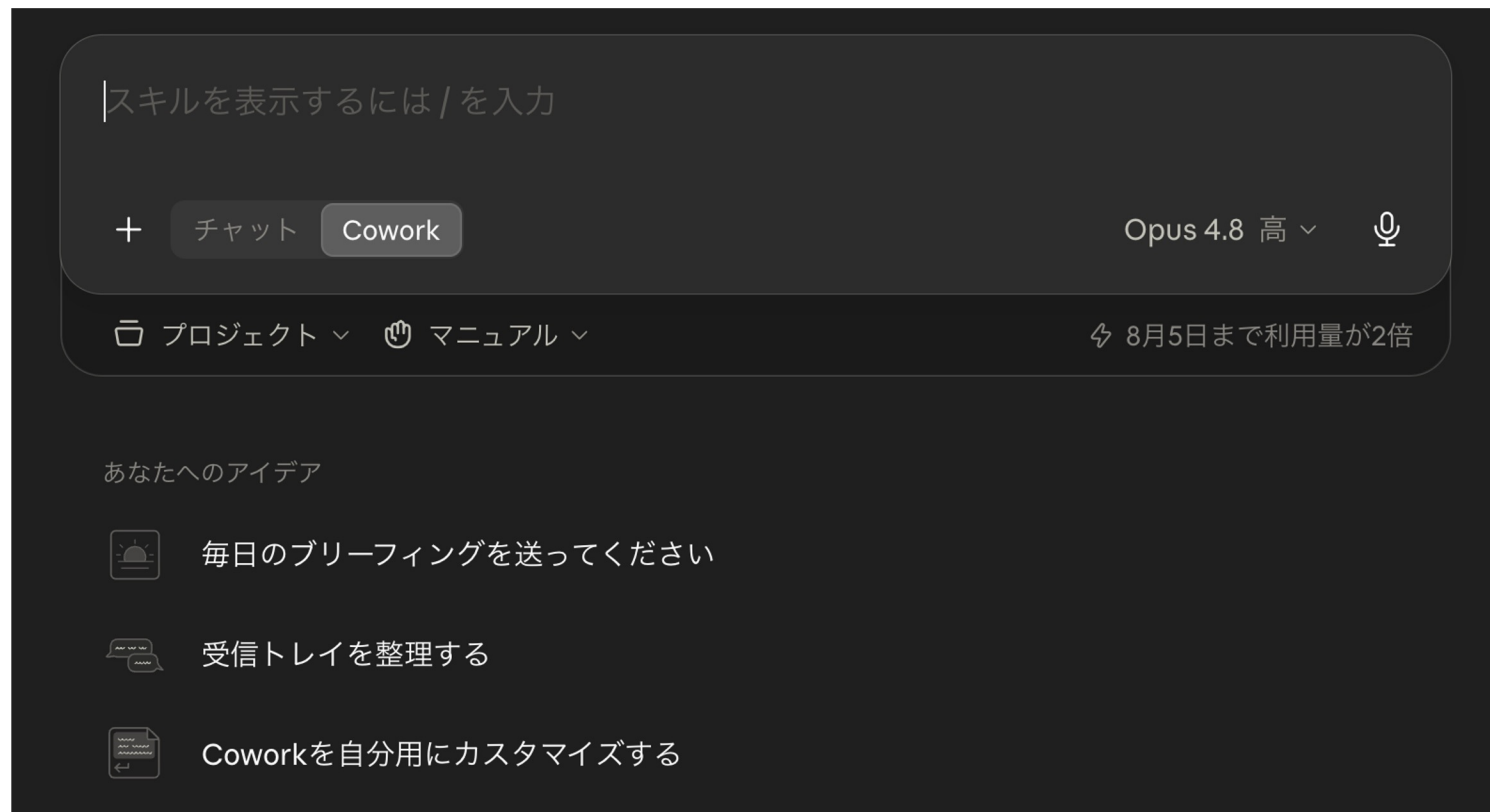
向いている人：デザイナー、PM、マーケティング担当者

特徴：Claude Codeへのハンドオフが可能

国内活用事例

- ・ LP ・ Webサイトの1日制作（ソウゾウ公式事例）
- ・ プレゼン資料・スライド自動生成（クリエイティブ企業）

Claude Cowork（ブラウザ・デスクトップアプリ）



Claude Cowork

非エンジニア向け業務自動化エージェント。ファイル操作が得意。

主な用途：資料整理、レポート作成、業務ワークフロー自動化

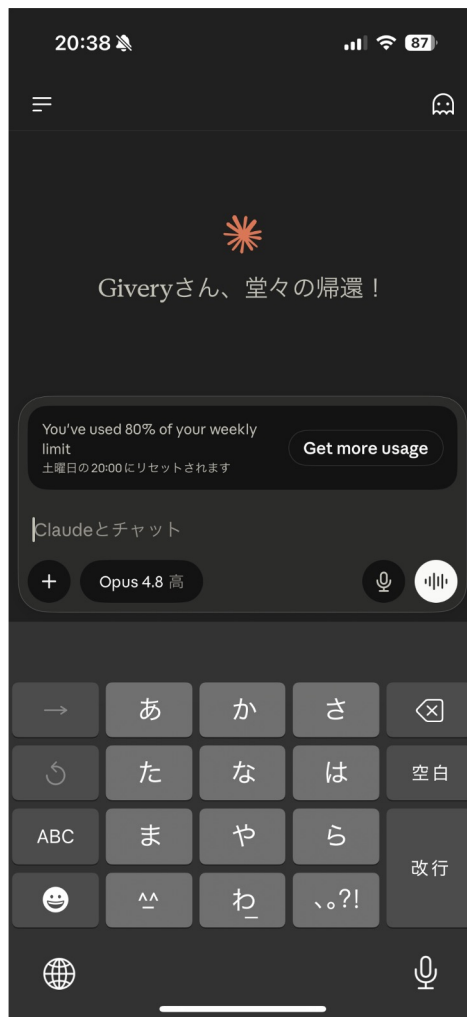
向いている人：ビジネスパーソン、知識労働者

苦手：高度なプログラミング

国内活用事例

- ・領収書フォルダからの経費精算自動化（Rimo社）
- ・契約書・報告書の自動整理（バックオフィス部門）

Claude スマホアプリ



Claude スマホアプリ

iOS/Android版のモバイルClaude。

主な用途：外出先での会話、音声入力、簡単リサーチ

向いている人：移動が多い人

苦手：大規模作業や精密作業

国内活用事例

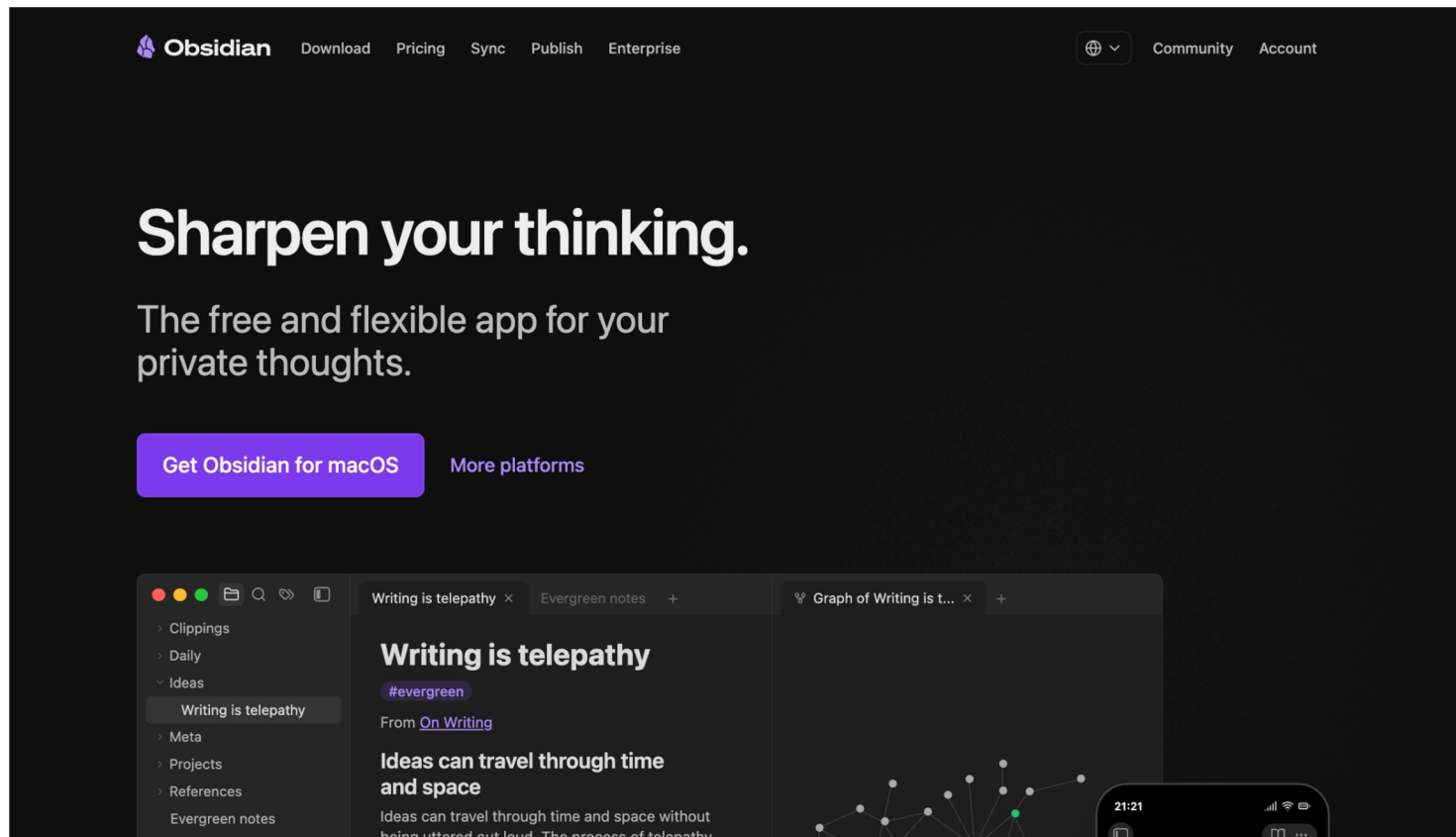
- ・ 移動中のメール作成・アイデアメモ（ビジネスパーソン）
- ・ 現場での資料確認と音声相談

Claude製品比較一覧

インターフェース	一言で概要	主な強み	向いている人	主な用途例	国内活用事例例	使い分けのポイント
Claude.ai (ブラウザ)	標準Webチャット	誰でもすぐ使える、 Projects/Artifacts充実	初心者〜一般ユーザー	文章作成、アイデア出し、調べ物	マーケティング調査・ 広告コピー作成	まずはここから。軽い作業全般
Claude Desktop (アプリ)	PC専用統合アプリ	Voice・Extensions・モード切替	ヘビーユーザー	日常業務全般	経費精算書自動作成	メイン環境として推奨
Claude Code (CLI)	ターミナル開発エージェント	Git・コマンド実行・大規模コード対応	ソフトウェアエンジニア	コード開発・リファクタリング	バグ修正・テスト自動化	本格的な開発作業
Claude Code (ブラウザ/Desktop)	GUI付きコードエージェント	ターミナル不要で視覚的操作	GUIを好む開発者	コード作成・編集集	Webアプリのプロトタイプ開発	開発者の中級者向け
Claude Code (VSCode等)	IDE内拡張機能	リアルタイム補完・編集	日常的にVSCodeを使う人	コーディング中の支援	API開発の高速化	実際にコードを書く場面
Claude Design	ビジュアルデザイン専用	プロトタイプ・スライド作成	デザイナー・PM・ marketer	UIデザイン・資料作成	LP/Webサイトの1日制作	デザイン・ビジュアル作業
Claude Cowork	非エンジニア向け業務エージェント	ファイル操作・業務自動化	ビジネスパーソン・事務職	資料整理・レポート自動化	領収書からの経費精算自動化	プログラミング不要の業務効率化
Claude スマホアプリ	モバイル版	音声入力・いつでも利用	移動が多い人	外出先の会話・メモ・リサーチ	移動中のメール作成・アイデアメモ	隙間時間・外出時

Obsidian活用 ベストプラクティス

Obsidianとは



Obsidianとは

Obsidianは、Markdownファイルベースのローカルファーストな個人知識管理（PKM）ツールです。

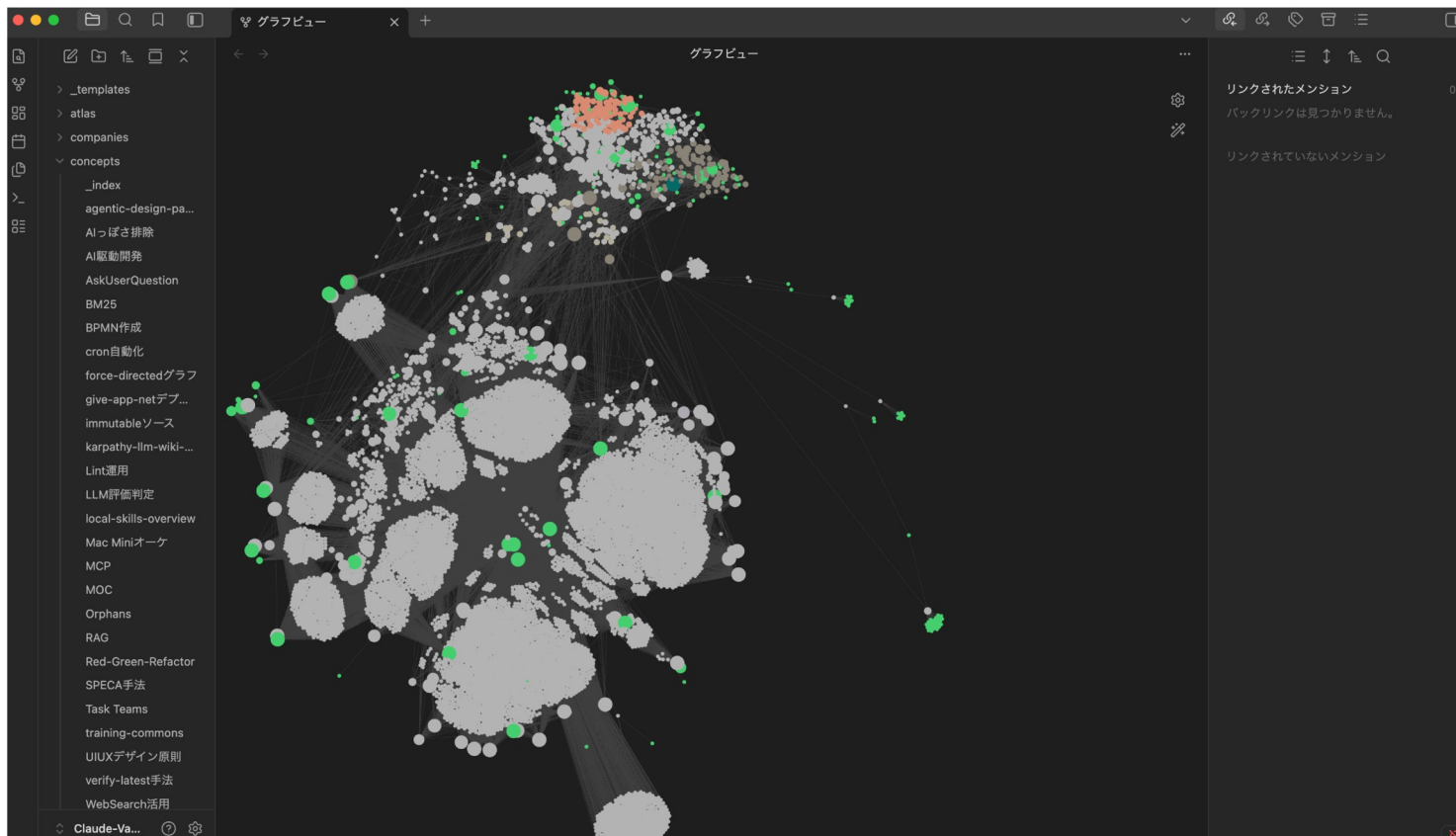
ノートを「Vault」（通常のローカルフォルダ）として保存し、双方向リンク、グラフビュー、バックリンクで知識をネットワーク状に繋げて拡張し、大規模なナレッジの**“生きた情報ネットワーク”の構築**が可能です。

オフライン完結、データ所有権が強く、無料（個人・商用）で利用可能。プラグインとテーマで高度にカスタマイズできます。

核心： 平文Markdownファイルでの管理 → アプリが消えても永続的に読める。

公式URL: <https://obsidian.md/>

Obsidianグラフビューのイメージ



Obsidianの強み

- データ所有権と将来性
 - ローカルMarkdownファイルなので、ロックインなし。Gitや任意の同期ツールでバックアップ可能。
- ネットワーク思考
 - 双方向リンクとGraph Viewで意外なつながりを発見。
- 拡張性
 - 数千のコミュニティプラグイン（AI統合、Kanban、Canvasなど） ↓ ↓
- 高速・プライベート
 - オフライン完結、クラウド不要。
- 柔軟性
 - Zettelkasten、PARA、MOCなど任意の方法論に対応



Obsidianの競合ツール

- Roam Research: 双方向リンクのパイオニア。有料、クラウド中心。
- Notion: データベース・コラボレーション強いが、ロックイン懸念。
- Logseq: Obsidianに似たオープンソースアウトライナー。
- その他: Evernote、OneNote、Tana、Capacities、RemNote。

Obsidianの優位性: ローカルMarkdown+プラグインの柔軟性で差別化。

Obsidianの競合ツール一覧

項目	Obsidian	Notion	Roam Research	Logseq	Tana	Evernote
データ形式	平文 Markdown（ローカルファイル）	独自クラウドDB	独自クラウドDB	平文 Markdown	独自構造	独自DB（エクスポート制限あり）
データ所有権	★★★★★（完全ローカル）	★★（クラウド依存）	★★（クラウド）	★★★★★（ローカル）	★★★（クラウド中心）	★★（エクスポートしにくい）
価格	無料（Syncは有料オプション）	無料プランあり / 有料	有料（月額）	完全無料	有料中心	無料プランあり / 有料
オフライン利用	完全オフライン	限定（同期必要）	限定	完全オフライン	限定	限定
双方向リンク	◎（強力 + Graph View）	○（データベース連携）	◎（元祖）	◎（アウトライナー重視）	◎	△
グラフビュー	◎（視覚的ネットワーク）	△	○	○	○	×
プラグイン/拡張性	★★★★★（数千のコミュニティプラグイン）	★★★（公式統合中心）	★★	★★★★（オープンソース）	★★★	★★
AI統合	◎（ローカル/クラウドLLM自由選択）	○（公式AI）	△	○	○	○
コラボレーション	△（Git/共有フォルダで可能）	◎（リアルタイム強力）	△	△	○	○
学習曲線	やや急（Markdown知識推奨）	緩やか	急	中程度	急	緩やか
向いている人	知識管理・長期保存・カスタム重視	チームコラボ・データベース	純粋な linked thinking	無料でObsidianライク	柔軟な構造化思考	シンプルメモ・検索
弱点	初期設定とコラボ	ロックイン・プライバシー	高額・クラウド依存	UIの好みがかかる	高額	古め・柔軟性不足

Obsidianの活用事例（国内）

日本でもビジネスパーソン（ITコンサル、エンジニア、弁護士、研究者）に人気です。

機密性が高い業務でローカル完結が評価されています（クライアント別知識ベース、提案書・レポート作成、社内Wiki代替など）

コンサルタントがCRMやプロジェクト管理にBases/Kanbanプラグイン活用している事例あります

→ <https://www.oflight.co.jp/en/columns/obsidian-knowledge-management-guide-2026>

企業内小規模チームでの共有Vault（Git同期など）事例も増加しています。

Obsidianの爆発的普及の背景



Andrej Karpathy (元OpenAI共同創業者・Director of AI)

2026年4月に「LLM Wiki」という知識管理パターンを公開しました。

- Markdownファイル (raw/フォルダに生データ、wiki/にLLMがコンパイルした構造化ページ) を基盤とし、Obsidianを「IDE/frontend」として明示的に推奨。
- Graph Viewで視覚化し、LLM (Claudeなど) が自動メンテナンスする「第二の脳」として活用。
- **Karpathy本人の発言例:**
「IDE: I use Obsidian as the IDE "frontend" where I can view the raw data, the compiled wiki, and the derived visualizations(Obsidianを、生データ・コンパイルされたwiki・そこから派生した各種ビジュアライゼーションを閲覧できるIDE的な"フロントエンド"として使っている)」

これがきっかけでObsidian + Claudeのワークフローが爆発的に流行しました (多数のチュートリアル・プラグインが存在)。

Vault（保管庫）の管理

Vault = 普通のフォルダ（.mdファイル群）。

- 複数Vault作成可能（例: 個人用/仕事用）。
- 同期：Obsidian Sync（有料）
iCloud/Dropbox/Gitなど。
- ベストプラクティス
定期バックアップ
.obsidianフォルダをgitignore（チーム共有時）
不要ファイル整理
ClaudeCodeとの自動連携
完全自動情報収集。

保管庫の構成例

```
Vault Root
├── Inbox/           (未整理キャプチャ)
├── Projects/        (アクティブプロジェクト)
├── Areas/           (継続領域: Health, Workなど)
├── Resources/       (参考: Books, Articles,
Wiki)
├── Journal/         (日誌: Daily, Weekly)
├── Archives/        (完了・古いもの)
├── Templates/
├── raw/             (AI入力用)
├── processed/       (AI処理後)
├── wiki/            (永続知識)
├── agents.md / CLAUDE.md (AI指示)
└── index.md         (全体MOC)
```

Obsidianに知識が蓄積する仕組み

さまざまな機能・拡張・連携により、Obsidianで管理するVault内に
人間・AIエージェントに関わる、あらゆる知識を連動的（Zettelkasten式）に管理可能です。

- Web Clipperプラグイン
- AIプラグイン（Smart Connections, Copilotなど）で自動要約・分類
- RSS / メール自動取り込み
- Templater + QuickAddでテンプレート自動適用
- Dataview / Tasksで動的クエリ
- LLMエージェントで定期整理（重複検知、リンク追加）

Zettelkastenとは

ドイツの社会学者ニクラス・ルーマンが考案した「カードボックス式」のメモ術・知識管理システムです。情報を集めるだけでなく、1つのメモに1つのアイデアを書き出し、メモ同士をリンクさせることで、脳のように思考やアイデアをネットワーク状に育てていく手法です

インストールと初期設定

配布フォルダ『Vault-starter』

「AIを第一の読み手として設計したObsidian Vault」です。

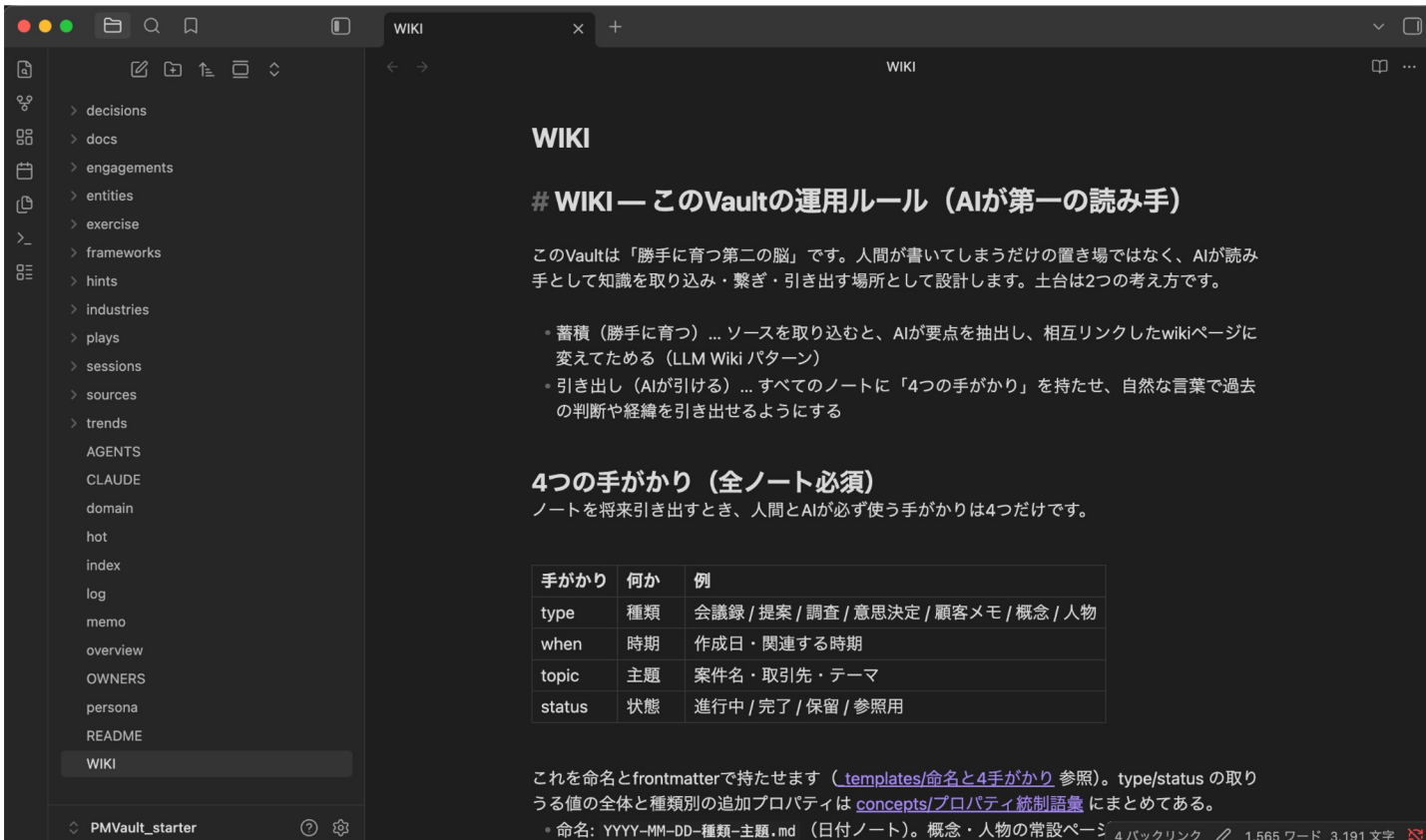
ふつうのメモ帳としてのObsidian Vaultと違い、人間が書くだけの置き場ではなく、資料を取り込むとAIが要点を抽出して相互リンクしたwikiページに変え、あとから自然な言葉で引き出せる形にためていきます。

プラグインもベクトルDBも使わず、Markdownだけで動く最小構成にしてあります（思想はKarpathyのLLM Wikiパターン、実装はAgriciDanielのclaude-obsidianを研修用に調整）。

- ・ **ためる（取り込むと自動でwiki化）**
- ・ **引き出す（4つの手がかりで過去の判断を時系列で返す）**

2つの機能で自律的に進化・更新される第2の脳

初期構成



フォルダ設計と運用思想

初期構成

- CLAUDE.md ...

起動時にAIへ渡す指示。「まず WIKI.md ・ index.md ・ hot.md ・ persona.md ・ domain.md を読め」と最初の視線を固定し

ます。文章ルール（結論から、売り文句を避ける）と、やってほしくないこと（.obsidianを触らない／削除は承認制／log.mdは追記のみ／機密は保存しない）もここ。

- AGENTS.md ...

CLAUDE.md の他ツール版。Claude以外のAIコーディングツールが読む同等の指示ファイル（AGENTS.md は複数ツール

共通の規約）。

- WIKI.md ...

運用ルールの本体。4つの手がかり、コアファイルの読み順、フォルダの役割、引き出し方、育て方、使える Skill、ADL

規約（公開情報のみ・一次情報URL＋日付・機密禁止）が1枚に。末尾に発展編（MOCの切り方、手法カタログ、コンサル層、チーム共有）を追記済み。

- index.md ...

マスター索引。全ページのカatalogで、AIはまずここを読んで必要なページだけ開く（トークン節約）。サブ索引に業界・事例・提案ストーリー・案件などの層を追加済み。

- hot.md ...

ホットキャッシュ。直近~500語の作業文脈。次のセッションを続きから始めるための短期記憶。

第2の脳としての基本層

- .raw/ ...

取り込み元の置き場。記事・議事メモ・資料をここに入れてingestする。
練習用に会議メモのサンプルが1本入っています。

- sources/ ...

取り込んだ資料から生成したwikiページ。

- concepts/ ...

テーマ・手法・考え方の常設ページ。ここに解説ライブラリを12枚追加済み（コンテンツ地図MOC／PKM手法カタログ／ノートの粒度／プロパティ統制語彙／AI第二の脳の設計パターン／共有の第二の脳／提案ストーリーの型／ピラミッド原則とSCQA／仮説思考／アナロジー設計の型／例え話見本／エグジビット・パターン）。

- entities/ ...

人・組織・トピックのページ。domain.mdは自分自身という特別なentity。

- sessions/ ...

/saveで残した会話アーカイブ。

コンサルとしての実務層

- engagements/ ...

案件ワークスペース。1案件1フォルダ。

- frameworks/ ...

分析フレームのライブラリ（3C見本つき）。

- benchmarks/ ...

再利用する数値の引き出し。

- plays/ ...

案件の進め方プレイブック。

- decisions/ ...

論点・前提・意思決定ログ。

- industries/ ...

業界プロファイル7本（製造・金融・小売流通・ヘルスケア製薬・エネルギー・公共・通信）。

- cases/ ...

事例バンク（一次情報で裏取りした種）。

- trends/ ...

四半期の動向スナップショット。

仕込んであるHook(命令の自動記録)

配布物で唯一の実行コード：.claude/hooks/log-prompt.mjs (Node製)

→役割は、『受講者がClaudeに命令を出すたびに発火し、その命令を[log.md](#)に1行追記する。』というものです。
手を動かすほど操作ログが自動で貯まる仕組みとなっています。

コードの動作

1. 標準入力からClaude Codeが渡すJSON (promptと作業ディレクトリcwd) を受け取る。
2. 命令文の言葉からカテゴリを自動判定する。正規表現で、調べ/リサーチ/ingest系→#調査、作成/生成/スライド/pptx系→#作成、修正/直して/fix系→#修正、引き出し/時系列/教えて系→#引き出し、それ以外→#その他。
3. YYYY-MM-DD HH:MM #カテゴリ [プロジェクト名] 命令本文 の形式でlog.mdに追記。命令が500文字を超えたら切り詰め。
4. [log.md](#)が無ければ、4手がかり付きのヘッダーを添えて新規作成する。

登録方法 (_setup/01...の手順)

プロジェクト個別ではなく、Claudeのユーザー設定 ~/.claude/settings.json のUserPromptSubmitに登録します。

理由

は明確で、別フォルダの演習プロジェクトで出した命令も、まとめてVaultのlog.mdに集めたいから。Vaultの場所は環境変数OBSIDIAN_VAULTで指定でき、既定は~/Documents/PMVault。

安全設計

書き込み先はlog.mdだけ、外部送信は一切なし。全体がtry/catch+exit 0で囲ってあり、Hookが失敗しても

Claudeの動

作は止まりません。研修で「Hookとは何か」を体で理解させるための、最小で無害な題材になっています。

その他の仕組み

type / when / topic / status ...それぞれ4つの判断基準

Vaultの背骨。全ノートのfrontmatterと命名（YYYY-MM-DD-種類-主題.md）に持たせ、これで過去の判断を絞り込む際

に使用します。中でもstatus（進行中／完了／保留／参照用）が一番重要です。

ingestループ ...

「勝手に育つ」要素の中心。rawに資料を入れてingestすると、

AIが概念・人物・出典を抽出して8~15枚のwikiページに分け、既存ページへ[[リンク]]を張り、[index.md](#)とlog.mdを更新

します。複製でなくリンクで繋ぐので、取り込むほどグラフが密になります。

.claude/settings.json の権限モデル ...

事故らないための安全枠です。Read/Glob/Grep/Edit/Write/WebSearch/WebFetchは自動許可、Bashは実行前に確認（

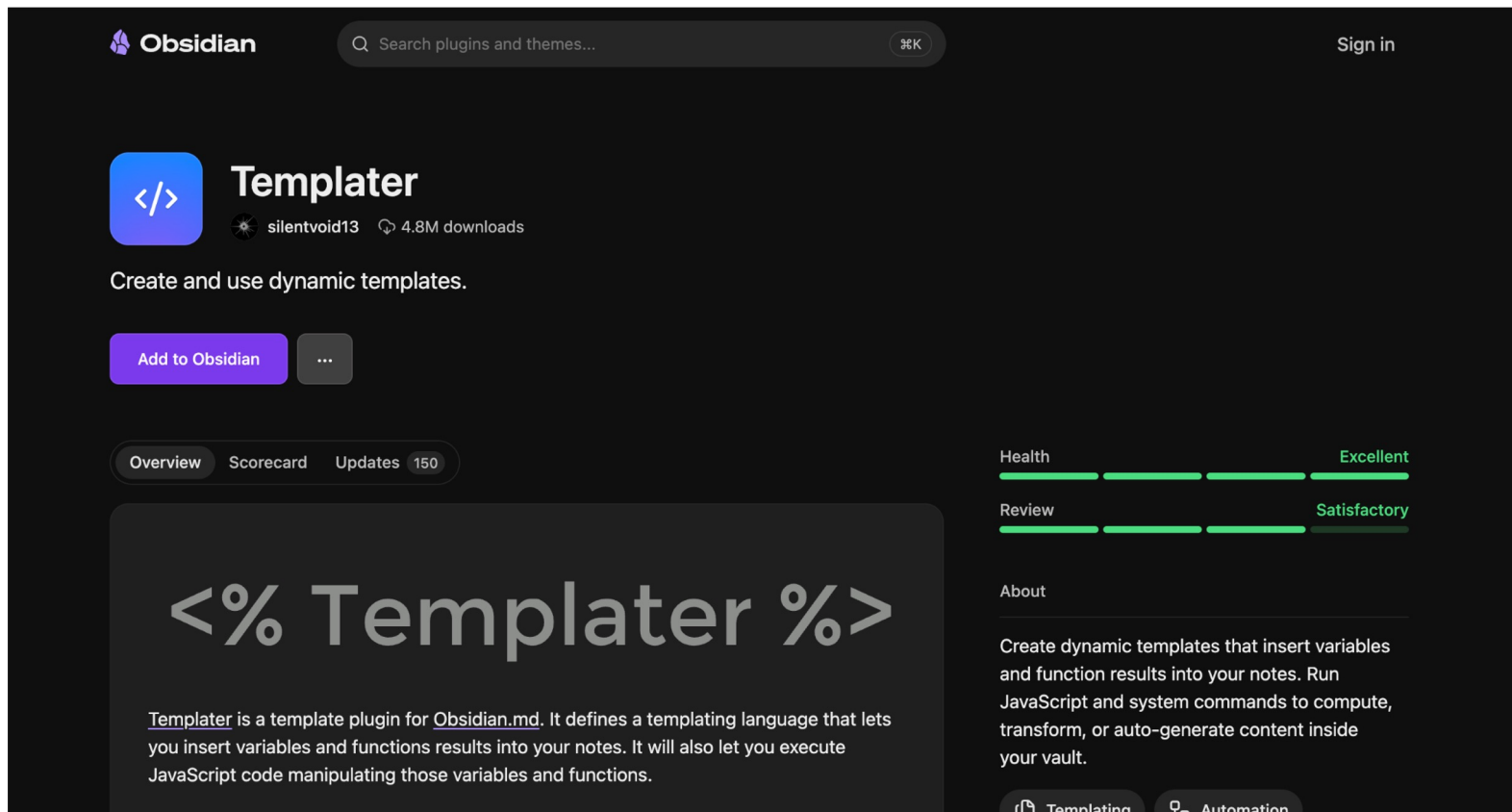
ask）、そして.obsidian配下は読み書きとも拒否（deny）。設定ファイルをうっかり壊させないためのガードです。

13のSkill（/名前呼び出し）...

- ためる: wiki（初期化・索引整備） / ingest（資料→wiki） / save（会話→ノート） / autoresearch（自律調査3ラウンド） / lint（健全性点検） / update-hot
- 引き出す: ask（過去の判断を時系列・論点で返す）
- 人間から引き出す: intake（楽に入力） / interview（暗黙知を深掘り）
- 自分用に育てる: tune-persona / update-domain
- 持ち出す: save-setup / load-setup（効いた設定を別案件へ再現）

必須プラグインと ショートカット

Templater



The screenshot shows the Obsidian plugin page for 'Templater'. At the top, the Obsidian logo and a search bar are visible. The plugin name 'Templater' is prominently displayed with its icon (a blue square with white code symbols). Below the name, the developer 'silentvoid13' and '4.8M downloads' are listed. A description states 'Create and use dynamic templates.' There are buttons for 'Add to Obsidian' and a menu icon. A tab bar shows 'Overview' (selected), 'Scorecard', and 'Updates' (150). The main content area features a large code snippet: `<% Templater %>`. Below this, a paragraph describes the plugin's functionality. On the right, a 'Health' section shows a green progress bar and the word 'Excellent'. Below that, a 'Review' section shows a green progress bar and the word 'Satisfactory'. An 'About' section contains a detailed description of the plugin's capabilities. At the bottom right, there are tags for 'Templating' and 'Automation'.

Obsidian

Search plugins and themes...

Sign in

Templater

silentvoid13 4.8M downloads

Create and use dynamic templates.

Add to Obsidian ...

Overview Scorecard Updates 150

`<% Templater %>`

Templater is a template plugin for Obsidian.md. It defines a templating language that lets you insert variables and functions results into your notes. It will also let you execute JavaScript code manipulating those variables and functions.

Health Excellent

Review Satisfactory

About

Create dynamic templates that insert variables and function results into your notes. Run JavaScript and system commands to compute, transform, or auto-generate content inside your vault.

Templating Automation

Templater（テンプレート/自動化）

Obsidian用のテンプレートプラグインで、**変数や関数の結果をノートに挿入できるテンプレート言語を定義**する。
さらに、それらの**変数や関数を操作するJavaScriptコードを実行**できる。 [obsidian](#)

つまり

普通のテンプレート（日付や見出しの自動挿入）を超えて、`<%tp.date.now()%>` のような動的な値や、JavaScriptによる自動処理を仕込むことができる。

→ デイリーノート生成、商談メモの雛形自動作成、フロントマターの自動埋め込みなどに向いています。

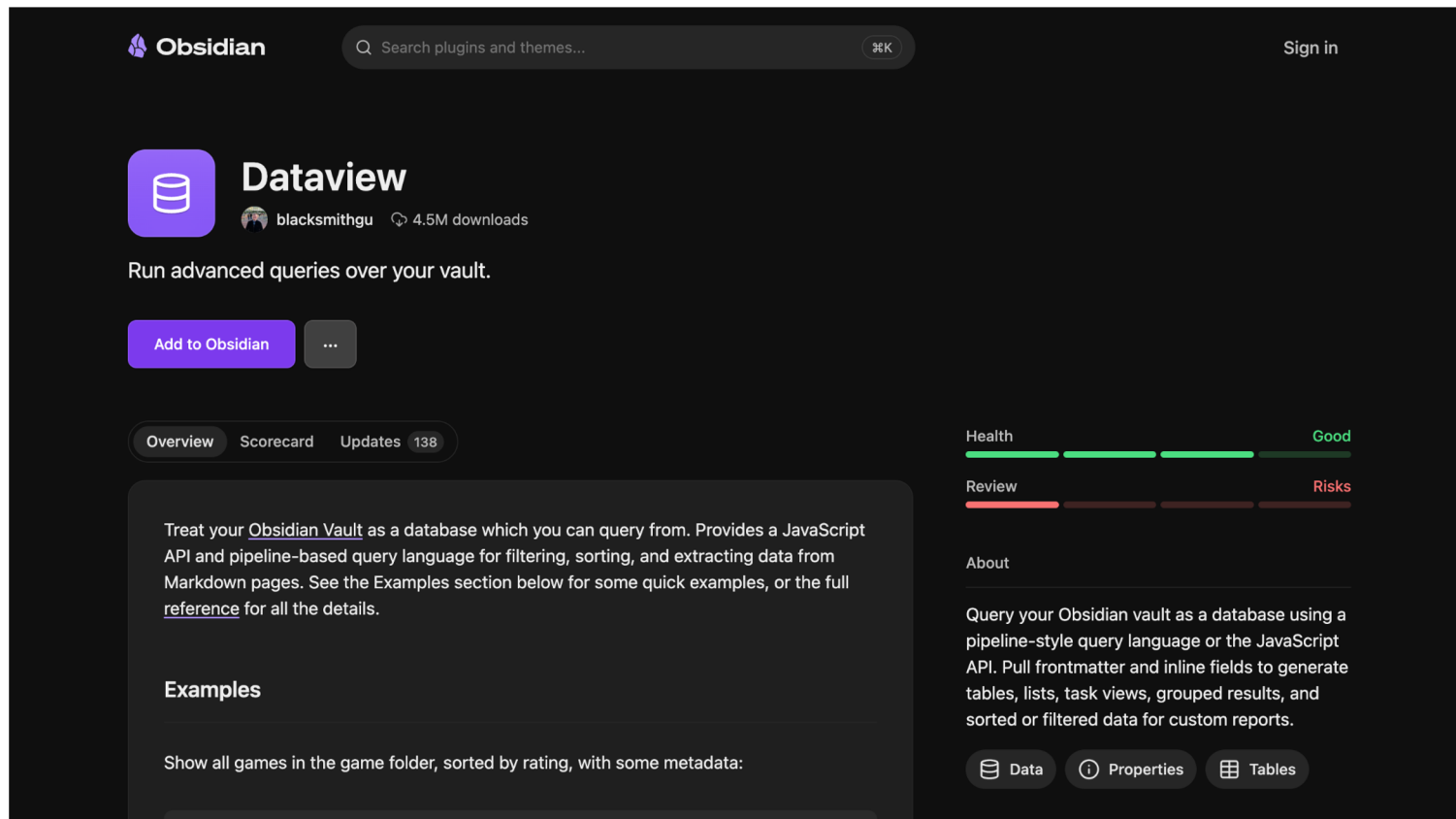
* 注意点

Templaterは任意のJavaScriptコードやシステムコマンドを実行できるため、信頼できないソースからのコード

の実行は危険なので、理解できる・信頼できるものだけを実行すべきとされている。

* 現在のバージョンは2.23.0、Obsidian 1.13.0以降対応、デスクトップ・モバイル両対応

Dataview



The screenshot shows the Obsidian Dataview plugin page. At the top, the Obsidian logo is on the left, a search bar in the center, and a 'Sign in' button on the right. The main header features the Dataview icon (a purple square with a white database cylinder), the name 'Dataview', the author 'blacksmithgu', and '4.5M downloads'. Below this is the tagline 'Run advanced queries over your vault.' and two buttons: 'Add to Obsidian' and a three-dot menu. A tab bar shows 'Overview' (selected), 'Scorecard', and 'Updates 138'. The main content area has a description: 'Treat your [Obsidian Vault](#) as a database which you can query from. Provides a JavaScript API and pipeline-based query language for filtering, sorting, and extracting data from Markdown pages. See the Examples section below for some quick examples, or the full [reference](#) for all the details.' Below the description is an 'Examples' section with the text 'Show all games in the game folder, sorted by rating, with some metadata:'. On the right side, there is a 'Health' status bar (green) labeled 'Good', a 'Review' status bar (red) labeled 'Risks', and an 'About' section with the text 'Query your Obsidian vault as a database using a pipeline-style query language or the JavaScript API. Pull frontmatter and inline fields to generate tables, lists, task views, grouped results, and sorted or filtered data for custom reports.' At the bottom right, there are three buttons: 'Data', 'Properties', and 'Tables'.

Obsidian

Search plugins and themes...

Sign in

 **Dataview**

 blacksmithgu 4.5M downloads

Run advanced queries over your vault.

Add to Obsidian ...

Overview Scorecard Updates 138

Treat your [Obsidian Vault](#) as a database which you can query from. Provides a JavaScript API and pipeline-based query language for filtering, sorting, and extracting data from Markdown pages. See the Examples section below for some quick examples, or the full [reference](#) for all the details.

Examples

Show all games in the game folder, sorted by rating, with some metadata:

Health Good

Review Risks

About

Query your Obsidian vault as a database using a pipeline-style query language or the JavaScript API. Pull frontmatter and inline fields to generate tables, lists, task views, grouped results, and sorted or filtered data for custom reports.

Data Properties Tables

Dataview

Name	Time Played	Length	Rating
Outer Wilds	November 19th - 21st, 2020	15h	9.5
Minecraft	All the time.	2000h	9.5
Pillars of Eternity 2	August - October 2019	100h	9
Nier Automata	November 2019	30h	9

「Obsidianをデータベースとして使う」場合に中核となるプラグイン。

Obsidian Vaultをデータベースのように扱い、そこに対してクエリを実行できる。[obsidian](#)

データの取り込み方は2通りに対応。

Dataviewは、Markdownのフロントマターとインラインフィールドから情報を引き出してデータを生成するため、フロントマターはドキュメント冒頭の `---` で囲まれた任意のYAML、インラインフィールドは `Key:: Value` 構文でMarkdown中に直接メタデータを書くことができる。

クエリは4モード

- ① SQLに似たパイプライン型のDataview Query Language (DQL)
- ② Markdown中に直接埋め込むインライン式
- ③ Dataviewインデックスへのフルアクセスを持つ高機能なJavaScript API (DataviewJS)
- ④ インラインJS式。

*セキュリティ面での注意

JavaScriptクエリは強力だが他のプラグインと同じアクセスレベルで動作し、ファイルの書き換え・作成・削除やネットワーク通信が可能。一方、通常のDataviewクエリはサンドボックス化されており、Vaultに悪影響を与えることはできない（その代わりに機能は限定的）。

Omnisearch

The screenshot shows the Obsidian plugin page for Omnisearch. At the top, the Obsidian logo is on the left, a search bar with the text "Search plugins and themes..." is in the center, and a "Sign in" link is on the right. The main header for the plugin features a purple square icon with a white magnifying glass, the name "Omnisearch" in large white text, the author's name "Simon Cambier" with a small profile picture, and "1.5M downloads" with a download icon. Below this, a description reads "Intelligent search for your notes, PDFs, and OCR for images." There are two buttons: a purple "Add to Obsidian" button and a grey button with three dots. A tabbed interface shows "Overview" as the active tab, with "Scorecard" and "Updates" (showing 160 updates) as other options. The main content area has a dark background with a trophy icon and text stating "Winner of the 2023 Gems of the Year in the 'Existing plugin' category". Below this, it says "Omnisearch is a search engine that 'just works'." and "It always instantly shows you the most relevant results, thanks to its smart weighting algorithm". On the right side, there are two progress bars: "Health" is at the "Excellent" level, and "Review" is at the "Satisfactory" level. Below these is an "About" section with the text: "Search your vault instantly with a BM25-ranked engine that prioritizes matches in filenames, headings, and content to surface the most relevant results. Index PDFs, Office docs and images via text extraction, tolerate typos, filter by file type or quoted expressions, and insert [[links]] or navigate results with keyboard-first".

Obsidian

Search plugins and themes...

Sign in

 **Omnisearch**

 Simon Cambier  1.5M downloads

Intelligent search for your notes, PDFs, and OCR for images.

Add to Obsidian ...

Overview Scorecard Updates 160

 Winner of the [2023 Gems of the Year](#) in the "Existing plugin" category 

Omnisearch is a search engine that "just works".

It always instantly shows you the most relevant results, thanks to its smart weighting algorithm.

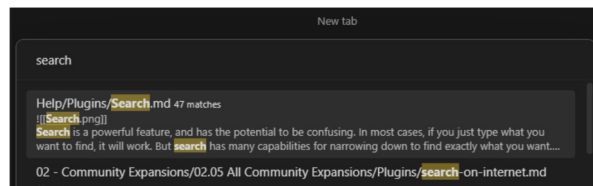
Health Excellent

Review Satisfactory

About

Search your vault instantly with a BM25-ranked engine that prioritizes matches in filenames, headings, and content to surface the most relevant results. Index PDFs, Office docs and images via text extraction, tolerate typos, filter by file type or quoted expressions, and insert [[links]] or navigate results with keyboard-first

Omnisearch



「とにかく動く」検索エンジンで、スマートな重み付けアルゴリズムにより、常に最も関連性の高い結果を即座に表示する。（2023年のGems of the Yearの「既存プラグイン」部門受賞作。）

[obsidianobsidian](https://obsidianobsidian.com)

主な特徴

ノート・Officeドキュメント・PDF・画像を高速に検索できる（画像/ドキュメント/PDFのインデックス化はText
Extractor経由）。BM25アルゴリズムによる自動スコアリングで、クエリ語がドキュメント本文・ファイル名・見出しに現れる回数に応じて関連度が決まっています。

使い勝手として、キーボードファーストでコアプラグインの「Quick Switcher」に似たワークフロー、タイプミスへの耐性、"引用符での完全一致" や -除外 の指定、.jpg や .md でのファイルタイプ絞り込み、検索結果から直接 [[link]] を挿入する機能などで有効です。

*注意

標準のObsidian検索も含め、日本語・中国語のような分かち書きしない言語は苦手です。中国語のサポートには追加プ
ラグインが必要と明記されており、日本語も同様に検索精度が落ちやすいので、導入後に日本語検索の挙動は必ず確認
してください。

Advanced Tables

The screenshot shows the Obsidian plugin page for 'Advanced Tables'. At the top, the Obsidian logo and a search bar are visible. The plugin's icon, a purple square with a white grid, is next to the title 'Advanced Tables' by 'tgroisinger', which has '3M downloads'. Below this is the description 'Improved table navigation, formatting, and manipulation.' and a purple 'Add to Obsidian' button. A tab bar shows 'Overview' (selected), 'Scorecard', and 'Updates 51'. The main content area lists features: Auto formatting, Excel-like table navigation, Spreadsheet formulas, Add/remove/move columns and rows, Set column alignment, Sort rows, Export to CSV, and Works on Obsidian Mobile. On the right, a 'Health' section shows a green progress bar at 'Excellent' and a 'Review' section at 'Passed'. An 'About' section describes the plugin's functionality. At the bottom, three buttons labeled 'Tables', 'Editing', and 'CSV' are shown.

Obsidian

Search plugins and themes...

Sign in

 **Advanced Tables**

 tgrosinger 3M downloads

Improved table navigation, formatting, and manipulation.

Add to Obsidian

Overview Scorecard Updates 51

Add improved navigation, formatting, and manipulation to markdown tables in Obsidian:

- Auto formatting
- Excel-like table navigation (tab/enter between cells and rows)
- Spreadsheet formulas!
- Add, remove, and move columns and rows
- Set column alignment (left, center, right)
- Sort rows by a specified column
- Export to CSV
- Works on Obsidian Mobile (See notes below)

Health Excellent

Review Passed

About

Edit and navigate Markdown tables like a spreadsheet using Tab/Enter cell and row movement; add, remove, and move columns and rows, and set column alignment. Use spreadsheet formulas, sort rows by any column, and export tables to CSV with mobile support.

Tables Editing CSV

Advanced Tables

URLは [table-editor-obsidian](#) ですが、正式名称は **Advanced Tables**。

Markdownテーブルのナビゲーション・整形・操作を改善するプラグインです。 [obsidian](#)

機能としては、自動整形、Excelのようなテーブル操作（Tab/Enterでセル・行を移動）、スプレッドシート数式、列や行の追加・削除・移動、列の配置指定（左・中央・右）、指定列でのソート、CSVエクスポート。モバイルでも動作します。

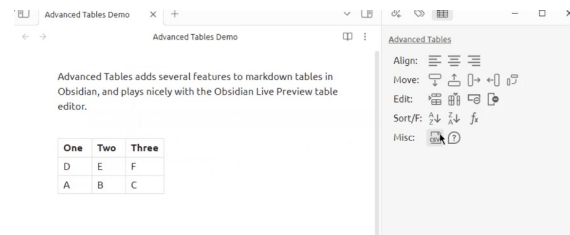
操作方法

| を1文字入力し、最初の見出しを打って Tab を押すと表が作られる。
カーソルが表内にあるとき、Tab で次のセル、Shift+Tab で前のセル、Enter で次の行、
Ctrl+Shift+D でテーブル操作用サイドバーが開く。

＊注意として、実験的な機能で不安定な場合があり、現在のノートのデータを削除してしまうバグの可能性もあるため、バックアップを取るようにしましょう（講師も1度発生しました）

Markdownの表は手書きだと崩れやすいので、

Dataviewで「読む」テーブルを作りつつ、手入力の生データ表はAdvanced Tablesで「書く／整える」
という役割分担になります。



ClaudeCode × リサーチmd化ハンズオン

ClaudeCode基礎

AI research and products that put safety at the frontier

AI will have a vast impact on the world. Anthropic is a public benefit corporation dedicated to securing its benefits and mitigating its risks.



Claude Opus 4.5

Introducing the best model in the world for coding, agents, computer use, and enterprise workflows.

Learn more

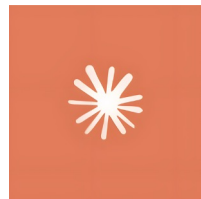
Introducing Claude Opus 4.5



Advanced tool use on the Claude Developer Platform



Anthropic



- **設立**：2021年
- **創業者**：元OpenAIの研究者たち（特に兄弟の**Dario Amodei**がCEO、**Daniela Amodei**が社長）
- **本社**：アメリカ・サンフランシスコ
- **ミッション**：「**安全で信頼できるAI**」を作ること
 - 「AIが人類に長期的に良い影響を与える」ことを最優先に、危険な挙動を極力抑えたAIを研究・開発している会社です。
 - 他のAI会社（例：OpenAI）と比べて「**AIの安全性（AI safety）**」を一番強く主張している点が特徴。

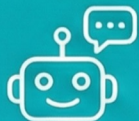
公式サイト引用

「Anthropicは、信頼性が高く、解釈可能で、制御しやすいAIシステムを構築するAI研究会社です。」



元OpenAIメンバー設立のAI企業

2021年設立。「信頼できるAIシステム」の研究開発を目指す。



主力製品はAI「Claude」

対話、要約、コーディングなど幅広いタスクをこなす大規模言語モデル。



独自の「憲法AI」アプローチ

AIが原則（憲法）に従い自己修正することで、安全性を確保する。



公益を追求する「PBC」

利益と社会的利益の両立が法的に義務付けられた公益目的会社。



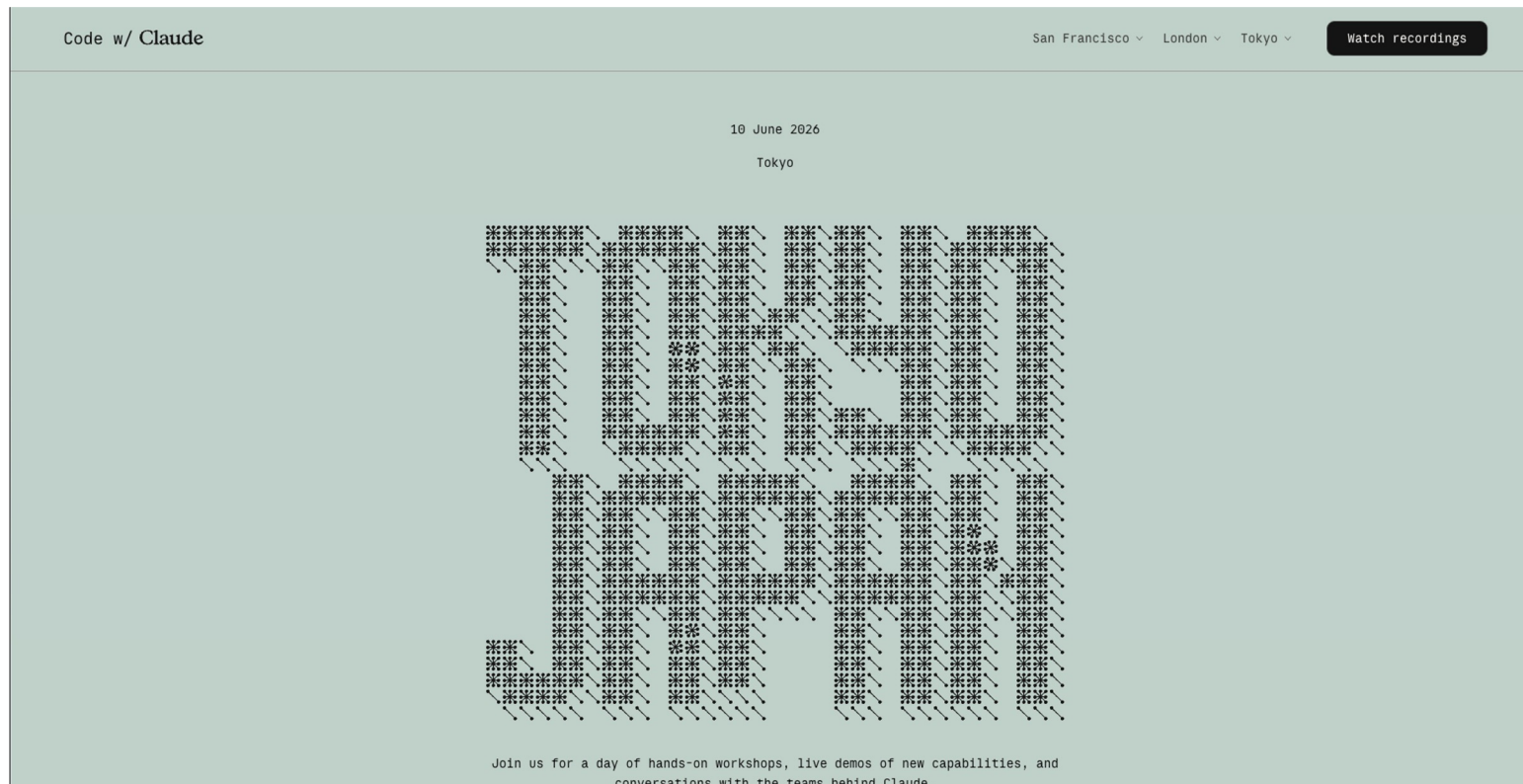
大手IT企業が巨額投資

AmazonやGoogleなどから数十億ドル規模の出資を受け、開発を加速。

公式発信の歩き方

Anthropicの公式発信は最先端生成AI知識の宝庫

Event : 年に1回 (2026年は6/10に東京で実施)



Code with Claudeは各国主要都市で毎年場所を変えて開催

サンフランシスコ(2024)→ロンドン(2025)→東京(2026)

Code w/ Claude

San Francisco ▾ London ▾ Tokyo ▾

Watch recordings

Watch recordings

Pick a track and watch the talks, as fresh as the day they were presented.

Q Search recordings

SESSION TRACK ▾

- ☐ Research
- ☐ Claude Platform
- ☐ Claude Code



Opening keynote

- Boris Cherny
- Ami Vora, Anthropic
- Dianne Penn, Anthropic
- Angela Jiang, Anthropic
- Katelyn Lesse, Anthropic
- Cat Wu, Anthropic



What's new in Claude Code

- Dickson Tsai, Anthropic



Memory and dreaming for self-learning agents

- Mahesh Murag, Anthropic



Caching, harnesses, and advisors: Building on Claude at GitHub scale

- Mario Rodriguez, GitHub



The expanding toolkit

- Lucas Gonzalez, Anthropic



How to get to production faster with Claude Managed Agents

- Jess Yan, Anthropic

YouTube公式アカウント

The screenshot shows the YouTube channel page for Anthropic. At the top, there's a navigation bar with the YouTube logo, a search bar, and icons for creating content, notifications, and the user's profile. Below the navigation bar, a large banner reads "Build with Claude" with the Anthropic logo. The channel name "Anthropic" is prominently displayed, along with the handle "@anthropic-ai", subscriber count "72.3万人", and video count "171 本の動画". A description follows: "We're an AI safety and research company. Talk to our AI assistant Claude on claude...さらに表示". Below this is a link to "anthropic.com" and a "チャンネル登録" (Subscribe) button. The channel's video tab is selected, showing a video titled "世界のソフトウェアを保護するための取り組み | プロジェクト..." (Efforts to protect the world's software | Project...). The video thumbnail features a man with glasses, and the video description lists several partner organizations: Amazon Web Services, Anthropic, Apple, Broadcom, Cisco, CrowdStrike, Google, JPMorgan Chase, Linux Foundation, Microsoft, NVIDIA, and Palo Alto Networks. The video has 426,681 views and was uploaded 3 months ago.

YouTube JP

検索

ホーム ショート 登録チャンネル

マイページ >

履歴 再生リスト 後で見る 高く評価した動画 オフライン

YouTube の他のサービス

- YouTube Premium
- YouTube Music
- YouTube Kids

探索

- 音楽
- 映画とテレビ番組

Build with Claude

Anthropic

@anthropic-ai · チャンネル登録者数 72.3万人 · 171 本の動画

We're an AI safety and research company. Talk to our AI assistant Claude on claude...さらに表示

anthropic.com、他 1 件のリンク

チャンネル登録

ホーム 動画 ショート ライブ コース 再生リスト

世界のソフトウェアを保護するための取り組み | プロジェクト...

426,681 回視聴 · 3 か月前

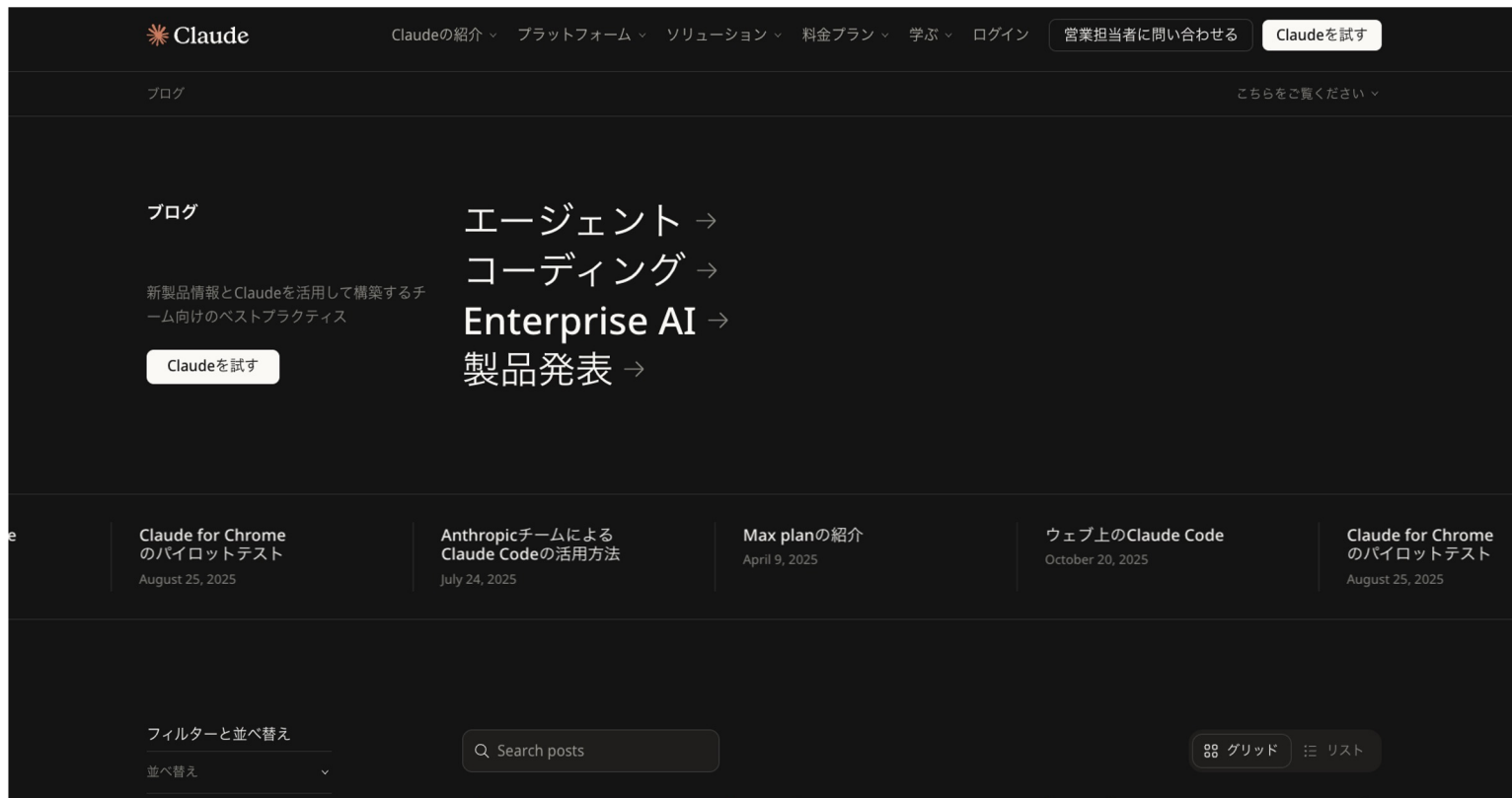
プロジェクト・グラスウィングは、世界で最も重要なソフトウェアのセキュリティ確保を目指し、Amazon Web Services、Anthropic、Apple、Broadcom、Cisco、CrowdStrike、Google、JPMorgan Chase、Linux Foundation、Microsoft、NVIDIA、Palo Alto Networksが結集した新たな取り組みです。

プロジェクト・グラスウィングを立ち上げたのは、Anthropic...

詳細

that they depend upon suddenly is less secure.

Blog：内部エンジニアによる解説などが中心



Newsroom : 最新発表や機能解説

ANTHROPIC

Research Economic Futures Commitments Learn News Try Claude

Newsroom

Press Inquires

press@anthropic.com

Non-media Inquiries

support@anthropic.com

Media assets

Download press kit



Announcements Sep 29, 2025

Introducing Claude Sonnet 4.5

Claude Sonnet 4.5 sets new benchmark records in coding, reasoning, and computer use while being Anthropic's most aligned model, accompanied by the release of the Claude Agent SDK for building capable agents

Product Oct 15, 2025

Introducing Claude Haiku 4.5

Claude Haiku 4.5 matches state-of-the-art coding capabilities from months ago while delivering unprecedented speed and cost-efficiency for complex tasks.

Announcements Sep 2, 2025

Anthropic raises \$13B Series F at \$183B post-money valuation

Anthropic raised \$13 billion in a Series F round at a \$183 billion valuation to expand enterprise offerings, safety research, and international growth as revenue grew from

Engineer : 技術目線での発表や解説

ANTHROPIC

Research

Economic Futures

Commitments ▾

Learn ▾

News

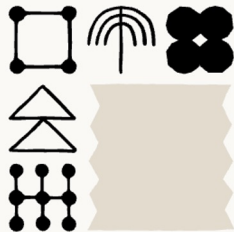
Try Claude



Engineering at Anthropic: Inside the team building reliable AI systems

Start building

Developer docs



Featured

Designing AI-resistant technical evaluations

What we learned from three iterations of a performance engineering take-home that Claude keeps beating.

Demystifying evals for AI agents

Jan 09, 2026

Effective harnesses for long-running agents

Nov 26, 2025

Introducing advanced tool use on the Claude Developer Platform

Nov 24, 2025

『Skillsを解説』の記事を読んでみましょう

 Claude

Claudeの紹介 ▾ プラットフォーム ▾ ソリューション ▾ 料金プラン ▾ 学ぶ ▾ ログイン

営業担当者に問い合わせる Claudeを試す

Blog / Skillsを解説 : Skillsはプロンプト、Projects、MCP、サブエージェントとどう違うのか

こちらをご覧ください ▾



Skillsを解説 : Skillsはプロンプト、Projects、MCP、サブエージェントとどう違うのか

SkillsはカスタムAIワークフローやエージェントを作成するための強力なツールとしてますます重要になっていますが、Claudeのスタック内ではどこに当てはまるのでしょうか？どのツールをいつ使用すべきか、そしてそれらがどのように連携するかについて説明していきましょう

🔍 カテゴリ

[エージェント](#)

📦 製品

Claudeのアプリ

Claude Developer Platform

📅 日付

2025-11-13

🕒 読了時間

5分

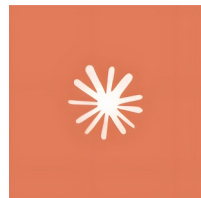
🔗 共有

[リンクをコピー](#)

Claude

特徴、強み、弱み

Claude



Anthropicの顔と言えば**Claude**というチャットAI（大規模言語モデル）です。

2026年現在、主な最新モデルは以下のような感じです（順に賢さ・能力が高い順）：

- **Claude Opus 4.5** → 最強モデル。コーディング・エージェント・長時間作業（30時間連続で自律作業可能）が特に優秀。オフィス作業（Excel・スライド・深いリサーチ）も得意。
- **Claude Sonnet 4.5** → バランス型。速度・コスト・性能のバランスが良く、企業で一番使われている。
- **Claude Haiku系** → 軽量・高速モデル（まだ4.x系まで進化中）

特徴：

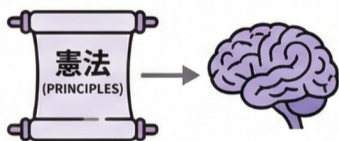
- 安全性が高い（有害な回答をかなり避ける）
- 長い文章・複雑な指示をよく理解する
- ツール使用（ブラウザ操作・コード実行・ファイル操作）が非常に強い
- **Claude Code**（ターミナルで動く本格コーディングエージェント）が爆発的に人気

Claude



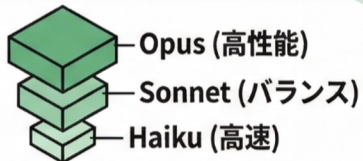
安全性を重視するAnthropic社が開発

元OpenAIのメンバーによって設立され、安全で信頼性の高いAIシステムの研究開発を目的としています。



「憲法 AI」によるトレーニング

AIが原則（憲法）に基づき、自らを「無害で役立つ」ように訓練する独自のアプローチを採用しています。



目的別のモデルファミリー

高性能な「Opus」、バランスの取れた「Sonnet」、高速な「Haiku」など、用途に応じたモデルが提供されています。



コーディング支援ツールも提供

開発者向けに、コーディングタスクを支援する強力なAIアシスタント「Claude Code」も展開しています。

Claude（ブラウザベースのAIチャット→絶賛進化中）



Tips : 人の投票によるAI性能評価『LMarena』

LMarena

New Chat

Leaderboard

Search



Please login now to save your chat history

If you have not yet created an account, please do so now so your chat history is not lost.

Login

Hide this

OverviewTextCodeVisionText-to-ImageImage EditSearchText-to-VideoImage-to-Video

Start Voting

Leaderboard Overview

See how leading models stack up across text, image, vision, and beyond. This page gives you a snapshot of each Arena, you can explore deeper insights in their dedicated tabs. Learn more about it [here](#).

Text20 hours ago

Rank	Model	Score	Votes
1	gemini-3-pro	1488	31,624
2	grok-4.1-thinking	1476	31,545
3	gemini-3-flash	1471	18,772
4	claude-opus-4-5-20251101-thi...	1468	23,402
5	claude-opus-4-5-20251101	1467	28,091
6	grok-4.1	1466	35,971
7	gemini-3-flash (thinking-min...	1464	13,318
8	gpt-5.1-high	1458	27,690
9	ernie-5.0-Q110	1454	7,732
10	claude-sonnet-4-5-20250929-t...	1451	41,942

Code4 days ago

Rank	Model	Score	Votes
1	claude-opus-4-5-20251101-thi...	1504	7,543
2	gpt-5.2-high	1475	1,691
3	claude-opus-4-5-20251101	1467	7,900
4	gemini-3-pro	1462	14,043
5	gemini-3-flash	1454	8,389
6	glm-4.7	1445	5,650
7	minimax-m2.1-preview	1414	7,201
8	gemini-3-flash (thinking-min...	1412	5,430
9	gpt-5.2	1399	1,632
10	gpt-5-medium	1397	3,929

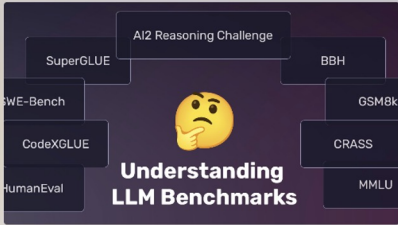
Tips : LLMの性能をタスク別ベンチマークで測る方法

 Arize AX ▾ Phoenix OSS Pricing Docs Learn ▾ Company ▾

Sign in [Book a demo](#) [Get started](#)

40 Large Language Model Benchmarks and The Future of Model Evaluation

Published April 11, 2025



With the accelerated development of GenAI, there is a particular focus on its testing and evaluation, resulting in the release of several LLM benchmarks. Each of these benchmarks tests the LLM's different capabilities—but are they sufficient for a complete real-world performance evaluation?

This blog will discuss some of the most popular LLM benchmarks for evaluating top models like GPT 4o, Gemma 3, or Claude. Further, we will discuss the LLMs' use in practical scenarios and whether these benchmarks are sufficient for complex implementations like agentic systems.

Evaluation Criteria: Traditional AI vs GenAI

Traditional AI algorithms, such as those used for classification, regression, and time-series forecasting, are typically deterministic systems. This means that for a specific set of inputs, the model is expected to produce a consistent output. While the model's predictions might deviate from

**Haziqa Said**
Contributor
[in](#) [X](#) [link](#)

[Evaluation Criteria: Traditional AI vs GenAI](#)

40 LLM Benchmarks

Understanding LLM Benchmarks

General Knowledge & Language Understanding Benchmarks

1. MMLU Benchmark
2. AI2 Reasoning Challenge
3. SuperGLUE

Tips : MMLU(Massive Multitask Language Understanding)

MMLUIは、様々な分野のタスクに対する言語モデルの性能を評価するためのベンチマーク（評価基準）です。

このベンチマークは、57種類の異なるタスクをカバーし、モデルがどれだけ多様な知識と問題解決能力を持っているかを評価します。

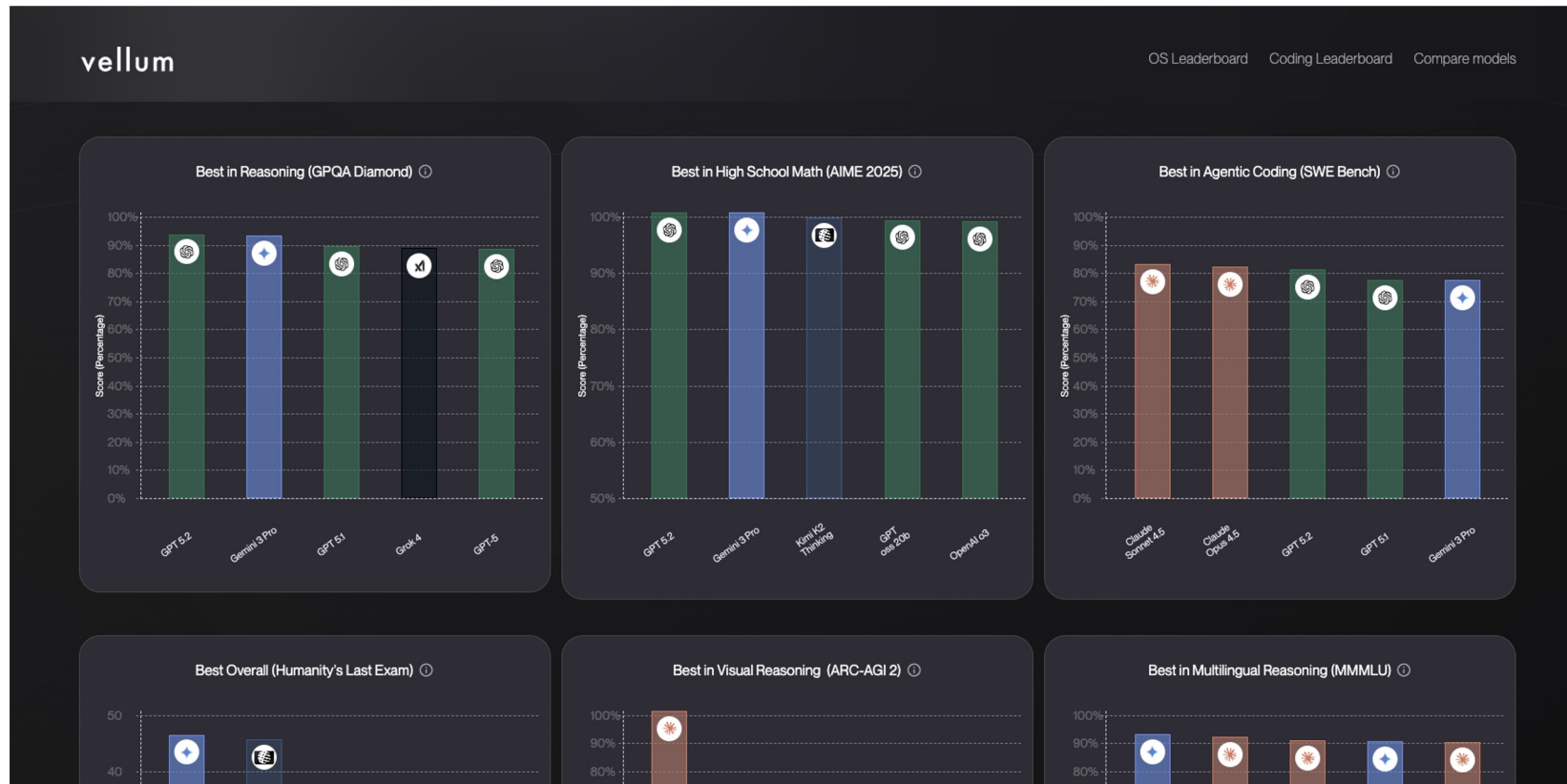
タスクには、数学、歴史、コンピュータサイエンス、法律などが含まれます。

具体的には、MMLUIは以下のような質問に対するモデルの回答を評価します：

- **歴史**：アメリカ独立宣言が署名された年は？
- **数学**： $x + 2 = 5$ を解いてください。
- **コンピュータサイエンス**：Pythonのforループの使い方を説明してください。

これらの質問を通じて、言語モデルがどれだけ正確に、そして迅速に正しい回答を導き出せるかを測定します。

Tips : vellumによる各ベンチマークの結果



Claudeの強み

- **安全性・倫理的設計が抜群（Constitutional AI）**

Anthropic独自の「憲法AI」アプローチにより、害のある出力や偏見を極力抑えています。ASL-3レベルの安全対策（CBRNリスク分類器など）が組み込まれており、高ステークスな業界（医療、金融、企業コンプライアンス）で最も信頼されやすいモデルです。他のモデル（GPT-5やGemini）と比べて「拒否率が高く、倫理的に慎重」な回答が特徴。

- **コーディング・ソフトウェアエンジニアリングで世界トップクラス**

SWE-Bench Verifiedで80.9%前後のスコアを記録し、2026年現在でもSOTA（State-of-the-Art）クラス。特にClaude Opus 4.5 / Sonnet 4.5は複雑なマルチファイル編集、バグ修正、エージェントワークフローに非常に強い。実務レベルのコードレビューや大規模リファクタリングで多くの開発者が「Claudeが一番」と評価しています。

- **非常に長いコンテキストウィンドウ（1Mトークン）**

大容量ドキュメント、長いコードベース、複数ファイルの同時解析に圧倒的に有利。Geminiと並んで長文処理の最強クラスで、研究資料の要約や大規模プロジェクトの分析で真価を発揮します。

- **自然で人間らしい文章・対話力**

出力が「人間らしい」「自然」「丁寧でニュアンス豊か」と評価されることが非常に多い。特にクリエイティブライティング、キャラクター描写、脚本、ストーリーテリング、長文の論理的展開でChatGPTやGeminiより優位と感じるユーザーが多数。ハルシネーション（幻覚）も比較的少なく、自己修正・事実確認能力が高い。

- **Artifacts / Claude Codeなどの生産性ツールが優秀**

コードレビュー、リアルタイム実行、Artifacts（生成物をインタラクティブに表示）などの機能が充実。開発ワークフローを劇的に効率化します。特にSonnet 4.5は「日常使いのベストバランス」と評判。

- **複雑な推論・ステップバイステップ思考が得意**

難しい数学、論理パズル、深い分析タスクで安定したパフォーマンスを発揮。特に「Plan Mode」や思考チェーンが強化された最新版は、長期タスクで強い。

Claudeの弱み

- **画像・動画生成ができない**

2026年現在もマルチモーダル出力（画像生成、動画生成）は非対応。GeminiやGPT-5シリーズに大きく劣るため、ビジュアルコンテンツ作成時は別ツールが必要になります。

- **速度とコストのバランスがモデルによって極端**

Opus系（最上位モデル）は非常に高価で遅め。Sonnet 4.5は速くてコスパ良いですが、Opusに比べて微妙に精度が落ちる場面あり。日常使いではSonnetを選ぶ人が多い一方、最高性能を求める人は予算が跳ね上がります。

- **メッセージ制限・使用量制限が厳しい**

Proプランでもピーク時に制限に引っかかりやすいという声が根強い。長時間連続使用や高負荷タスクで「途中で切られる」経験をするユーザーが一定数います。

- **リアルタイム検索・最新情報への弱さ**

ブラウジング機能はあるものの、GPT-5やGrokに比べて検索の正確性・鮮度で劣るとの評価あり。研究やニュース系タスクではPerplexityやGeminiの方が有利な場面が多い。

- **クリエイティブな「遊び心」やユーモアが控えめ**

安全性重視の副作用で、過度に慎重・真面目なトーンになりがち。GrokやChatGPTのような「楽しい・ウィットに富んだ」回答を求める人には物足りない場合があります。

- **一部のUI/フロントエンドタスクでGeminiに劣る**

視覚デザインやWebアプリの美しさではGemini 3 Proなどに負けるケースが報告されています。ロジック・バックエンドはClaudeが強いが、見た目の洗練度では差が出やすい。

コスト

Claudeのコスト

 Claude

Claudeの紹介 ▾ プラットフォーム ▾ ソリューション ▾ 料金プラン ▾ 学ぶ ▾ ログイン [営業担当者に問い合わせる](#) [Claudeを試す](#)

料金プラン [こちらをご覧ください](#)

料金プラン

個人 チーム&エンタープライズ API



無料
Claudeを試す

\$0
どなたでも無料でご利用いただけます

[Claudeを試す](#)

- ✓ ウェブ、iOS、Android、デスクトップ上でのチャット
- ✓ コード生成とデータの可視化
- ✓ 文章作成、編集、コンテンツ制作
- ✓ テキストと画像の分析
- ✓ ウェブ検索機能
- ✓ Create files and execute code
- ✓ デスクトップ拡張機能で、Claudeの機能をさらに解放



Pro
日々の生産性向上のために

\$17
年間サブスクリプション割引（\$200前払い）適用時の月額料金。月額請求の場合は\$20。

[Claudeを試す](#)

無料プランのすべての機能に加え、以下が含まれます。

- ✓ より多い使用量*
- ✓ ウェブとターミナルからClaude Codeにアクセス
- ✓ プロジェクトに無制限にアクセスし、チャットやドキュメントを整理
- ✓ リサーチへのアクセス
- ✓ Google Workspaceとの接続：メール、カレンダー



Max
Claudeを最大限に活用

\$100から
1人あたり月額請求額

[Claudeを試す](#)

Pro planのすべての機能に加え、以下が含まれます。

- ✓ Proの5倍または20倍の使用量の選択可能*
- ✓ すべてのタスクに対し高い出力制限
- ✓ Claudeの高度な機能への早期アクセス
- ✓ 混雑時の優先アクセス

Claudeのコスト：Team/Enterprise



Team plan

5名以上の小規模チーム向け

Team planを入手

Standardシート **\$25**

年次請求の場合はシートあたりの月額料金。月次請求の場合はシートあたり30ドルです。

プレミアムシート（より多くの使用量*） **\$150**

シートあたりの月額料金

- ✓ Claude Codeが含まれます
- ✓ Microsoft 365、Slackなどと連携
- ✓ 組織全体のエンタープライズサーチ



Enterprise plan

大規模なビジネス向け

営業担当者に問い合わせる

Team planのすべての機能に加え、以下が含まれます。

- ✓ より多い使用量*
- ✓ 拡張コンテキストウィンドウ
- ✓ Claude CodeはPremiumシートで利用可能
- ✓ Google Docs cataloging
- ✓ きめ細かなアクセス権限設定ができるロールベースアクセス制御
- ✓ クロスドメインID管理システム（SCIM）
- ✓ 監査ログ

Claudeのコスト：API

最新モデル

[営業担当者に問い合わせる](#)[開発を始める](#)

Opus 4.5

エージェントの構築とコーディングに適した最も優れたインテリジェントモデル

入力	\$5 / MTok
----	------------

出力	\$25 / MTok
----	-------------

プロンプトキャッシュ

書き込み	\$6.25 / MTok
------	---------------

読み取り	\$0.50 / MTok
------	---------------

Sonnet 4.5

インテリジェンス、コスト、スピードの最適なバランス

入力	
----	--

プロンプト：20万トークン以下	\$3 / MTok
-----------------	------------

プロンプト：20万トークン超	\$6 / MTok
----------------	------------

出力	
----	--

プロンプト ≤ 200K トークン	\$15 / MTok
-------------------	-------------

プロンプト：20万トークン超	\$22.50 / MTok
----------------	----------------

プロンプトキャッシュ

20万トークン以下

書き込み	\$3.75 / MTok
------	---------------

読み取り	\$0.30 / MTok
------	---------------

20万トークン超

書き込み	\$7.50 / MTok
------	---------------

読み取り	\$0.60 / MTok
------	---------------

Haiku 4.5

最速かつ最もコスト効率の高いモデル

入力	\$1 / MTok
----	------------

出力	\$5 / MTok
----	------------

プロンプトキャッシュ

書き込み	\$1.25 / MTok
------	---------------

読み取り	\$0.10 / MTok
------	---------------

Claudeのコスト：個人/Team/Enterprise

プラン	月額（目安）	内容
Free	\$0	まず試す用。基本機能
Pro	\$17（年間割引） / \$20（月額	Claude Code使える、使用量多い
Max	\$100～（5x/20x）	Proより使用量が多い
Team Standard	\$25/人（年） / \$30（月）	チーム管理・連携機能
Team Premium	\$150/人	多くの使用量と優先権
Enterprise	問い合わせ	大企業向け、拡張機能付き

Claudeのコスト：API

モデル名	入力トークン	出力トークン	用途
Opus 4.5	\$5 / 1Mトークン	\$25 / 1Mトークン	高性能・複雑タスク向け
Sonnet 4.5	\$3 / 1M	\$15 / 1M	バランス型（値段と性能の中間）
Haiku 4.5	\$1 / 1M	\$5 / 1M	コスパ重視（軽いタスク向け）
キャッシュ有効 時（読み取り/ 書き込み割引）	割引あり	割引あり	大量処理で節約可能

1Mトークン ≡ 約75万～100万語（概算）

中くらいの利用（例: 100回 / 日）
100t × 500t input + 1000t output
（入力:50,000t/出力:100,000t）

Sonnet 4.5の場合
入力：50k/1M×\$3 = \$0.15
出力：100k/1M×\$15 = \$1.5
→合計 ≡ \$1.65/日
→月30日 → 約 \$49.5/月

大量テキスト処理（例: 500回 / 日）
500t × 500t input + 1000t output
（入力：250,000t/出力：500,000t）

Sonnet 4.5の場合
入力：0.25M×\$3 = \$0.75
出力：0.5M×\$15 = \$7.5
→合計 ≡ \$8.25/日
→月30日 → 約 \$247.5/月

モデルごとの違い

Claudeの変遷

AnthropicのAI「Claude」は2023年の登場以来、安全性を重視した「憲法AI」を基盤に、急速な進化を遂げてきました。

2023年3月:
Claude 1 登場



2023年7月:
Claude 2
一般公開



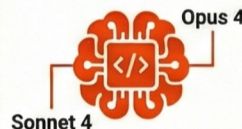
2024年3月:
Claude 3
ファミリー発表



2024年6月:
Claude 3.5
Sonnet



2025年5月:
Claude 4
世代へ



2025年11月:
Claude 4.5
Opus



Anthropic初の言語モデルとして、承認された一部ユーザー向けにリリースされました。

初めて一般ユーザーが利用可能になり、より長い文章の要約能力などが向上しました。

Opus、Sonnet、Haikuの3モデル構成で登場し、多くのベンマークでGPT-4を上回る性能を記録しました。

上位モデルのOpusを超える性能を達成し、コード生成などをリアルタイムでレビューできる新機能「Artifacts」を導入しました。

コーディングと論理的推論能力が飛躍的に向上したClaude 4 OpusとSonnet 4が登場しました。

コーディングと業務タスクの能力をさらに強化し、文脈長の制限を事実上なくす「無限チャット」機能を搭載しました。

Claudeの変遷

用途	Claudeが最も強いモデル	競合が勝つ場合の主な相手
コーディング・バグ修正	Opus 4.5 / Sonnet 4.5	GPT-5.2 Codex（安定性）
大容量ドキュメント解析	ほぼ全モデル（1Mトークン）	Gemini（2Mトークン拡張時）
自然な文章・クリエイティブ	Sonnet 4.5 / Opus 4.5	—（Claudeが優位）
画像/動画生成	×	Gemini / GPT-5
高速・低コスト日常使い	Sonnet 4.5	Gemini Flash / Grok
高ステークス・安全第一	Opus 4.5	—（Claudeが圧倒的）

使用上の注意

開発用途での生成AIの注意点

『勝手に直す』問題

- 意図以上に：
 - 構造変更
 - 命名変更
 - リファクタ を行うことが報告されています。

最低限の対策：

変更は最小限にしてください。
既存の構造・命名・設計方針は維持してください。

開発用途での生成AIの注意点

設計を聞かずにいきなり手を動かす

- いきなり書き始める
- 後から破綻

対策（超重要）：

まず設計案を箇条書きで提示してください。
私のOKが出るまで実装しないでください。

または /plan モードでしっかりと設計してから動作させる など

開発用途での生成AIの注意点



- ・実行環境未確認な『すぐに動作することを優先したコード』を出力
- ・テストを書かない（本番コードのみ出力）
- ・セキュリティ意識が低い（.env直書きしたりchmod判断が甘いなど）
- ・ログ、エラー処理が雑 などなど

対策

- ・人間と役割分担してタスクを割り振る
- ・SkillsやCommand、Rulesなどの『AIエージェントの育成』をしっかりと行う

ClaudeCodeを完全に人間の後輩と同じように育成する意識

知能はアインシュタイン。技術はDarioAmodei。ただし新卒。

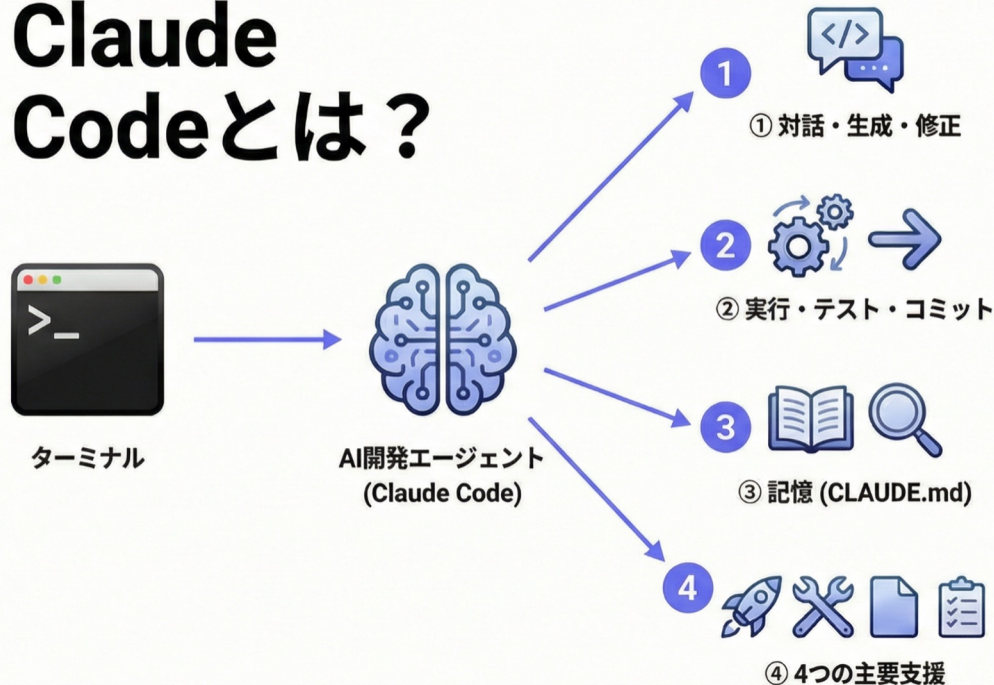
主要な機能

主要な機能（公式ドキュメントより）

主な機能	簡単な説明	公式参考URL / 詳細ページ
コード生成・機能実装	自然言語で「こんな機能を作って」と言うだけで、計画→コード作成→動作確認まで自動で進めてくれる	https://code.claude.com/docs/en/overview (Build features from descriptions)
デバッグ・バグ修正	エラーメッセージやバグの説明を貼るだけで、コードベース全体を見て原因特定→修正提案・適用	https://code.claude.com/docs/en/overview (Debug and fix issues)
コードベース理解・質問回答	プロジェクト全体の構造を把握して「この部分どうなってる?」「なぜこう設計した?」などに答える	https://code.claude.com/docs/en/overview (Navigate any codebase)
ファイル編集・操作	ファイルの読み書き・作成・削除を許可ベースで安全に実行（sandbox対応でセキュリティ強化）	https://www.anthropic.com/engineering/claude-code-sandboxing
Gitワークフロー自動化	コミット、ブランチ作成、PR作成、merge conflict解決、履歴検索などを自然言語で操作	https://www.anthropic.com/engineering/claude-code-best-practices (Git operations)
MCP（外部ツール連携）	GitHub、Google Drive、Figma、Slack、ブラウザ操作など外部サービスと接続して情報取得・操作	https://code.claude.com/docs/en/mcp
Skills（カスタムスキル）	自分やチームの専門知識・ワークフローをYAML+Markdownで定義してClaudeに「教える」機能	https://code.claude.com/docs/en/skills
Checkpoint / 巻き戻し	長時間タスク中に自動保存され、Esc2回 or /rewind で前の状態に簡単に戻せる	https://www.anthropic.com/news/enabling-claude-code-to-work-more-autonomously
Subagents / 並列作業	メインエージェントが専門タスクをサブエージェントに分担（例: フロントとバックエンド同時）	https://www.anthropic.com/news/enabling-claude-code-to-work-more-autonomously
VS Code拡張・Chrome連携	VS Code内で直接使えたり、Chrome拡張でブラウザをClaude Codeから操作可能（Beta）	https://www.anthropic.com/engineering/claude-code-best-practices + CHANGELOG
Sandboxing & パーミッション	危険操作を隔離・許可制にし、prompt injection対策で安全に自律動作させる	https://www.anthropic.com/engineering/claude-code-sandboxing
Background Tasks	devサーバーなど長時間実行プロセスをバックグラウンドで維持しながら他の作業継続	https://www.anthropic.com/news/enabling-claude-code-to-work-more-autonomously

ClaudeCodeとは

Claude Codeとは？



 **ターミナルで対話するAIエージェント**
自然言語の指示に基づき、コードの生成、デバッグ、リファクタリングなどを自律的に実行します。

 **開発ワークフローを「実行」**
ファイルの直接編集、テストコマンドの実行、Gitコミットまで、一連のタスクを自律的にこなします。

 **プロジェクトの文脈を「記憶」**
`CLAUDE.md` ファイルからコーディング規約やアーキテクチャを学習し、プロジェクトに最適化された支援を提供します。

 **4つの主要な開発支援**
①機能実装、②バグ修正、③コードベース理解、④定型タスク自動化をカバーします。

ClaudeCode チートシート

Claude Code 全設定チートシート

スラッシュ/CLI/settings/CLAUDE.md/Hooks/Skills/MCP/権限を1枚で / 2026-05 時点 公式仕様草稿

[illegible][illegible]

Sources (公式・一次情報/クリックで開く)

1. Commands (全スラッシュコマンド一覧) — code.iaude.com/docs/en/commands
2. CLI reference (コマンドとフラグ) — code.iaude.com/docs/en/cli-reference

4. Memory (CLAUDE.md / rules / auto memory) — code.claude.com/docs/en/memory

5. Skills (skill と custom command) — code.claude.com/docs/en/skills
6. Sub-agents (サブエージェント) — code.claude.com/docs/en/sub-agent

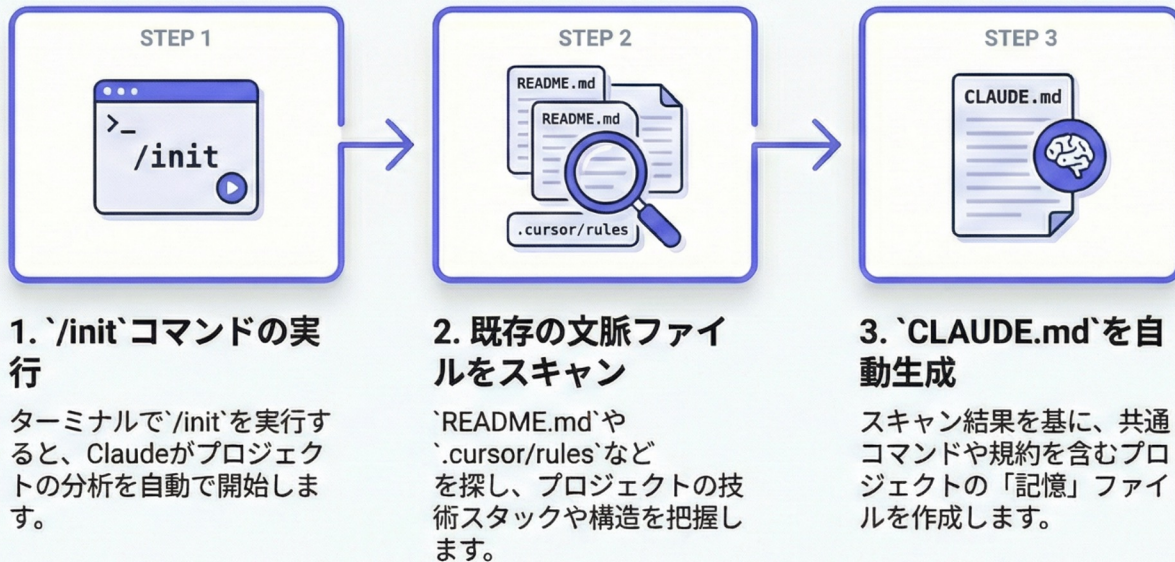
7. Hooks (フックイベント・type) — code.claude.com/docs/en/hooks

CLAUDE.md と その設定

コマンドとルール

/initコマンド

Claude Codeの`/init`コマンドは、プロジェクトの初期設定を自動化し、AIとの対話を効率化する第一歩です。



/initコマンド

ClaudeCodeを現在のプロジェクト（ディレクトリ）に対して**初期設定**する

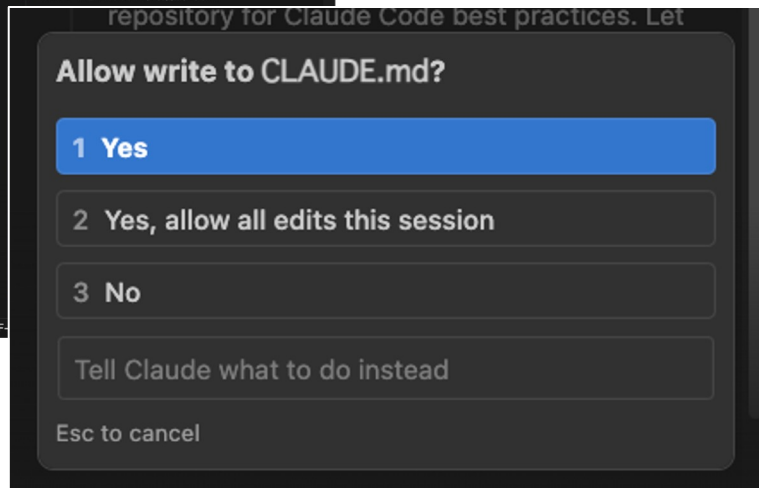
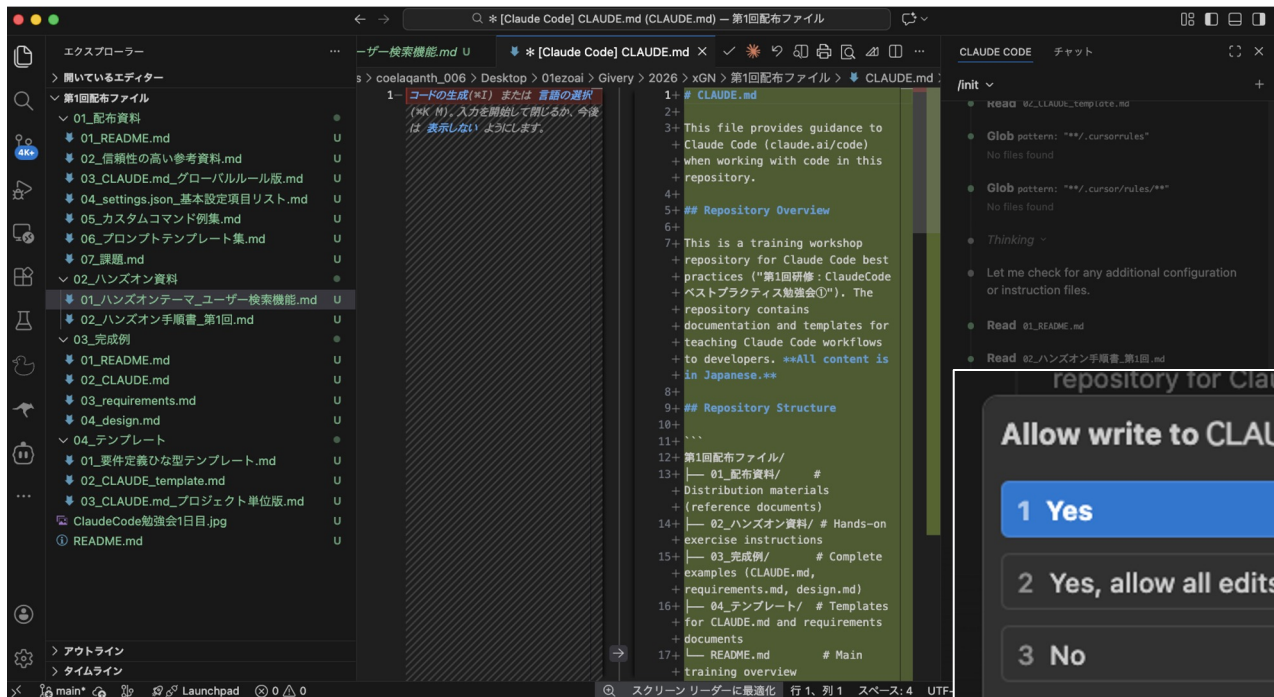
- **役割**

プロジェクトの構造をClaudeが理解しやすくするための設定ファイル（.claudecodeなど）を作成したり、プロジェクトのコンテキスト（何を作るためのコードか等）を読み取らせたりします。

- **使い方**

新しいプロジェクトでClaudeCodeを使い始める際に、最初に一度だけ実行します。

/initコマンドを実行する



/initコマンド実行結果

CLAUDE.md — 第1回配布ファイル — 追跡対象外

```
CLAUDE.md > *bc # CLAUDE.md
1  # CLAUDE.md
2
3  This file provides guidance to Claude Code
  (claude.ai/code) when working with code in this
  repository.
4
5  ## Repository Overview
6
7  This is a training workshop repository for Claude Code
  best practices ("第1回研修: ClaudeCodeベストプラクティス勉強会
  ①"). The repository contains documentation and templates
  for teaching Claude Code workflows to developers. **All
  content is in Japanese.**
8
9  ## Repository Structure
10
11  ...
12  第1回配布ファイル/
13  |— 01_配布資料/ # Distribution materials (reference
  | documents)
14  |— 02_ハンズオン資料/ # Hands-on exercise instructions
15  |— 03_完成例/ # Complete examples (CLAUDE.md,
  | requirements.md, design.md)
16  |— 04_テンプレート/ # Templates for CLAUDE.md and
  | requirements documents
17  |— README.md # Main training overview
18  ...
19
20  ## Key Documents
21
22  - [01_配布資料/03_CLAUDE.md_グローバルルール版.md] (01_配布資
  料/03_CLAUDE.md_グローバルルール版.md) - Global CLAUDE.md
  rules for organization-wide use
23
  - [04_テンプレート/02_CLAUDE_template.md] (04_テンプレ
  ート/02_CLAUDE_template.md) - Template for creating
  project-specific CLAUDE.md files
```

CLAUDE CODE チャット

/init

- Thinking
- I've created the CLAUDE.md file for this repository. Here's a summary of what it includes:
Key sections:
 - Repository Overview:** Explains this is a Japanese training workshop for Claude Code best practices
 - Repository Structure:** Maps the 4 main directories and their purposes
 - Key Documents:** Links to the most important reference materials (global rules, templates, hands-on instructions)
 - Build/Test Commands:** Notes this is documentation-only with no commands
 - Hands-on Theme:** Describes the example project (User Search Service API)
 - Workshop Flow:** Documents the recommended workflow taught in the training (/init → CLAUDE.md → requirements → design → review)
 - Important Conventions:** Guidance for maintaining file naming and language consistency

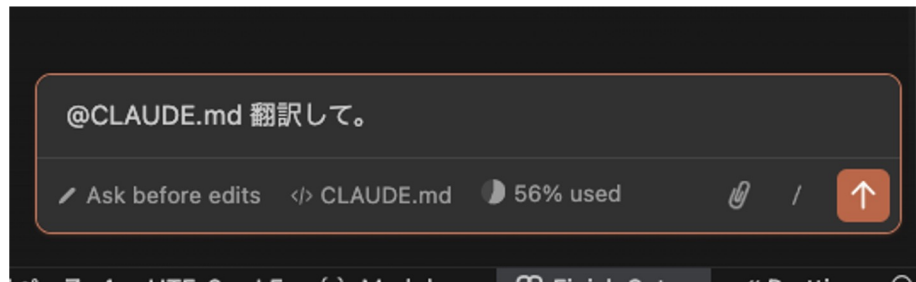
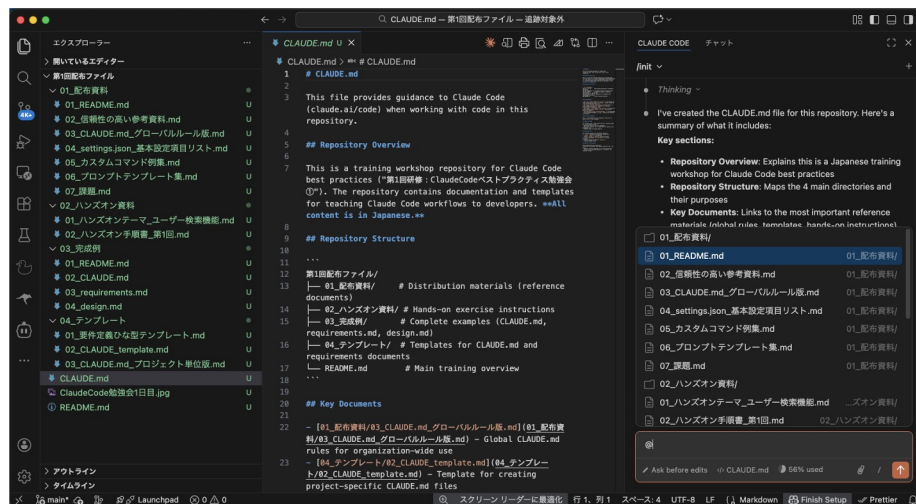
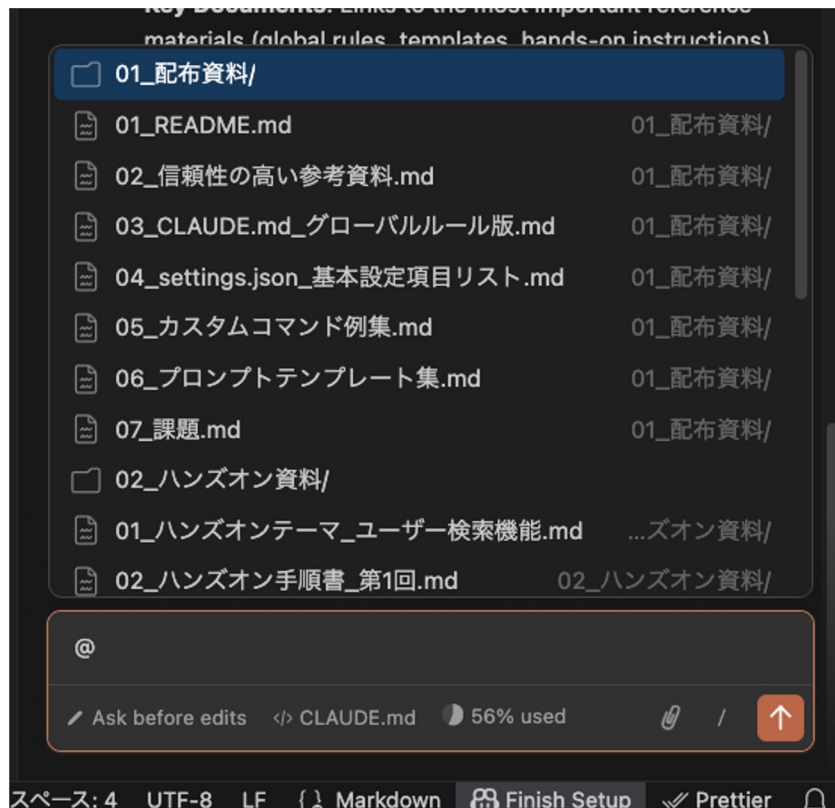
The file is concise and focused on what's essential for Claude Code to be productive in this training materials repository, without including generic development advice or repeating information that can be easily discovered by reading the files.

Esc to focus or unfocus Claude

Ask before edits CLAUDE.md 56% used

main* Launchpad 0 0 0 スクリーンリーダーに最適化 行 1、列 1 スペース: 4 UTF-8 LF Markdown Finish Setup Prettier

@からファイルを指定して動作



1 操作につき人間が確認するモードでの動作

The screenshot displays the Claude Code interface. On the left, a file editor shows the content of `CLAUDE.md` in Japanese. The text describes the file's purpose as guidance for Claude, mentions a repository, and lists key documents. A confirmation modal is overlaid on the right, asking "Make this edit to CLAUDE.md?". The modal has three options: "1 Yes", "2 Yes, allow all edits this session", and "3 No". Below these options is a text input field labeled "Tell Claude what to do instead" and a button labeled "Esc to cancel".

CLAUDE CODE チャット

/init

- Key Documents: Links to the most important reference materials (global rules, templates, hands-on instructions)
- Build/Test Comm with no command:
- Hands-on Theme Search Service AP
- Workshop Flow: I taught in the train requirements → di
- Important Conve naming and langu

The file is concise and Code to be productiv without including gen information that can I

@CLAUDE.md 日本語訳し

CLAUDE.md

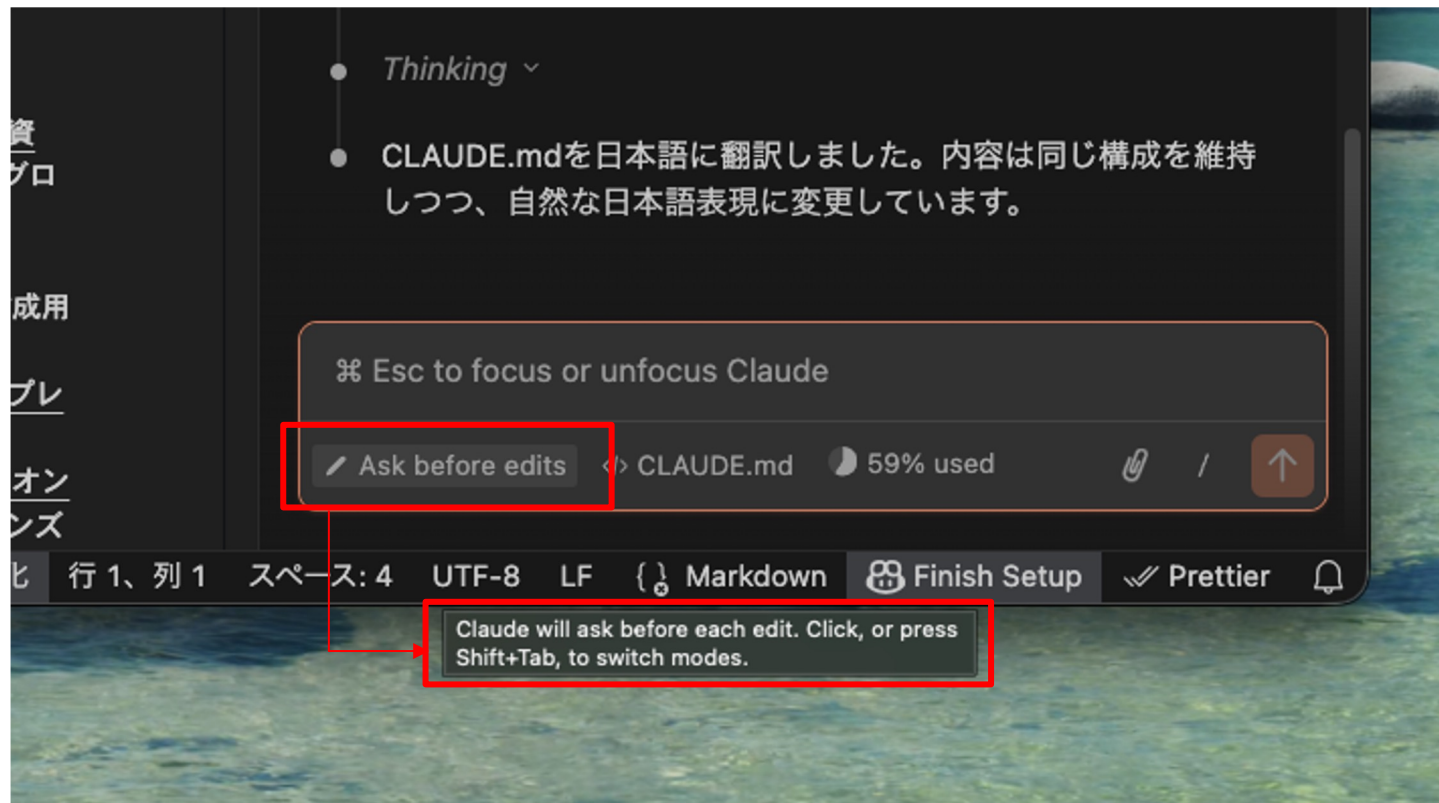
Make this edit to CLAUDE.md

- 1 Yes
- 2 Yes, allow all edits this session
- 3 No

Tell Claude what to do instead

Esc to cancel

Shift + Tabでモードの切り替えが可能



ClaudeCodeの基本3モード

ClaudeCodeを現在のプロジェクト（ディレクトリ）に対して**初期設定**する

- **Planモード**

基本的にファイルの作成や編集は行わない。

ユーザーとの対話の中で、情報整理や次のアクションをタスク化する

- **Edit Automaticallyモード**

人間の承認は最低限でファイル操作を行う。

初めてのコマンドなどは承認が入るが、基本的には自律的に動作

- **Ask before editsモード**

AIチャット上で完結する操作は自律的に動作するが、実際のファイル操作は行わない
ファイル操作を行う前には人間に承認を得る

デフォルトコマンド

その他デフォルトコマンド

コマンド	概要（一言で）	主な目的・役割	使用タイミング	メリット	注意点・落とし穴	ベストプラクティス例
<code>/init</code>	プロジェクト初期化	CLAUDE.md作成 + プロジェクト理解の基盤作り	新規プロジェクト開始時、リセットしたい時	最初から正確な提案が得やすくなる	既存のCLAUDE.mdを上書きする可能性	プロジェクトルートで最初に必ず実行
<code>/add</code>	ファイル・ディレクトリをコンテキストに追加	必要なファイルをClaudeに認識させる	話の途中で「あのファイルも見て」とき	コピペ不要で正確な編集が可能	追加しすぎるとトークン超過	<code>/add @src/main.py</code> のように具体的に指定
<code>/edit</code>	コードを編集提案・適用	選択範囲やファイルを直接修正	バグ修正、リファクタ、機能追加時	高速＆高品質な変更、diffで確認可	大きな変更は事前に計画確認を	編集前にコードを選択 → <code>/edit</code> → 具体指示
<code>/cmd</code>	ターミナルコマンド実行	<code>npm install</code> , <code>git commit</code> などをClaude経由で	依存インストール、ビルド、git操作時	IDE内で完結、Claudeが安全なコマンド提案	危険コマンド（ <code>rm -rf</code> など）に注意	<code>/cmd npm run build</code> のように明確に
<code>/clear</code>	会話履歴を完全クリア	コンテキストをリセット	話題が変わった時、応答が遅くなった時	トークン節約・応答速度向上	重要な会話が消える	1タスク終わるごとに使う習慣を
<code>/context</code>	現在のトークン使用量を表示	コンテキストの消費状況を確認	応答が遅い時、大量追加した後	オーバーフロー回避に必須	表示するだけ（自動調整はしない）	追加作業のたびにチェック

その他デフォルトコマンド

コマンド	概要（一言で）	主な目的・役割	使用タイミング	メリット	注意点・落とし穴	ベストプラクティス例
<code>/undo</code>	最後の変更を元に戻す	誤編集の即時リポート	適用した編集が気に入らなかった直後	安心して試せる	複数ステップ戻せない場合あり	編集適用後にすぐ内容確認 → 必要なら即 <code>/undo</code>
<code>/ask</code>	質問・説明モード	コードの解説や理由を聞く	「なぜこう書くの?」「この部分の意味は?」	学習・理解が深まる	編集はしてくれない	<code>/ask</code> このアルゴリズムの時間計算量は?
<code>/debug</code>	デバッグ支援	エラー原因特定・修正提案	ランタイムエラー・バグ発生時	ステップごとの原因追及が得意	再現コードが必要	エラーメッセージ+関連コードを選択して実行
<code>/test</code>	テストコード生成	ユニットテスト・統合テスト作成	新機能実装後・リファクタ前後	テストカバレッジが劇的に上がる	テスト対象のコードをしっかり認識させる	<code>/test src/utils/date.js</code> のように対象指定
<code>/review</code>	コードレビュー	改善点・バグ・スタイル指摘	PR作成前・コード完成時	人間より客観的・網羅的	好みがプロジェクト規約とズレる場合あり	変更範囲を選択 → <code>/review</code>
<code>/refactor</code>	リファクタリング提案	コードの構造・可読性・パフォーマンス改善	スパゲッティコード・古いコード整理時	大規模整理が短時間で可能	意図しない動作変更のリスク	小さな範囲から始め、diffをしっかり確認
<code>/model</code>	使用モデル切り替え	Claude 3.5 Sonnet Opus など変更	複雑タスク時・速度優先時	タスクに最適なモデル選択可能	Opusは高コスト	<code>/model claude-3-5-sonnet-20241022</code> のように指定

CLAUDE.md

CLAUDE.mdとは

The screenshot displays the Claude Code IDE interface. The left sidebar shows a file explorer with a tree structure of files and folders. The main editor window shows the content of the `CLAUDE.md` file, which is a configuration file for Claude Code. The file is organized into sections: `# CLAUDE.md`, `## リポジトリ概要` (Repository Overview), `## リポジトリ構成` (Repository Structure), and `## 主要ドキュメント` (Main Documents). The `## リポジトリ概要` section describes the file's purpose and provides a list of files and their roles. The `## リポジトリ構成` section shows the directory structure. The `## 主要ドキュメント` section lists the main documents and their locations. The right sidebar shows a chat window with the text `/init` and a list of instructions for using Claude Code, including `Hands-on Theme`, `Workshop Flow`, and `Important Conventions`. The bottom status bar shows the current file is `main*`, the editor is in `Launchpad` mode, and the file encoding is `UTF-8`.

CLAUDE.md

```
# CLAUDE.md
このファイルは、Claude Code (claude.ai/code) がこのリポジトリで作業する際のガイダンスを提供します。

## リポジトリ概要
これはClaude Codeベストプラクティスの研修用リポジトリです（第1回研修：ClaudeCodeベストプラクティス勉強会①）。開発者にClaude Codeのワークフローを教えるためのドキュメントとテンプレートが含まれています。

## リポジトリ構成
...
第1回配布ファイル/
├─ 01_配布資料/      # 受講者向け配布資料（参考資料）
├─ 02_ハンズオン資料/ # ハンズオン演習の手順書
├─ 03_完成例/        # 完成例（CLAUDE.md、requirements.md、design.md）
├─ 04_テンプレート/   # CLAUDE.mdと要件定義のテンプレート
├─ README.md         # 研修全体の概要
...

## 主要ドキュメント
- [01_配布資料/03_CLAUDE.md_グローバルルール版.md] (01_配布資料/03_CLAUDE.md_グローバルルール版.md) - 組織全体で使用するグローバルルール
- [04_テンプレート/02_CLAUDE_template.md] (04_テンプレート/02_CLAUDE_template.md) - プロジェクト固有のCLAUDE.md作成用テンプレート
- [04_テンプレート/01_要件定義ひな型テンプレート.md] (04_テンプレート/01_要件定義ひな型テンプレート.md) - 要件定義テンプレート
- [02_ハンズオン資料/02_ハンズオン手順書_第1回.md] (02_ハンズオン資料/02_ハンズオン手順書_第1回.md) - ステップバイステップのハンズオン手順
```

CLAUDE CODE チャット

/init

- **Hands-on Theme:** Describes the example project (User Search Service API)
- **Workshop Flow:** Documents the recommended workflow taught in the training (`/init` → `CLAUDE.md` → `requirements` → `design` → `review`)
- **Important Conventions:** Guidance for maintaining file naming and language consistency

The file is concise and focused on what's essential for Claude Code to be productive in this training materials repository, without including generic development advice or repeating information that can be easily discovered by reading the files.

@CLAUDE.md 日本語訳して

CLAUDE.md

- Thinking
- **Edit CLAUDE.md**
Modified
- Thinking
- CLAUDE.mdを日本語に翻訳しました。内容は同じ構

⌘ Esc to focus or unfocus Claude

Ask before... CLAU... 59% used

CLAUDE.mdとは

Claude Code専用の特別な設定ファイル（Markdown形式）。

Claude Codeは毎回のセッション開始時に自動的にこのファイルを読み込み、プロジェクトの「永続的な記憶（persistent memory）」として扱います。

これにより、毎回同じ説明を繰り返さなくても、Claudeがプロジェクトのルールや構造を理解した状態で作業を始められるようになります。

→ Claude Codeにとっての

「プロジェクト憲法」 「オンボーディング資料」 「常時読み込まれる最上位プロンプト」

CLAUDE.mdに記載すべき項目

優先度	カテゴリ	具体例・記載すべき内容	なぜ重要か（効果）
★★★★★	プロジェクト概要	プロジェクト名・目的・ドメイン・主要機能の1行〜3行まとめ	Claudeが「何を作っているのか」を最初に把握
★★★★★	技術スタック	使用言語、フレームワーク、主要ライブラリ、DB、インフラなど（バージョンも可能な限り）	間違った技術選定を防ぐ
★★★★☆	ディレクトリ構造	src/, tests/, docs/ などの主要フォルダの役割を簡潔に説明	大規模リポジトリで迷子になりにくくなる
★★★★☆	コーディング規約	命名規則、インデント、import順、async/await vs Promise、named export優先など具体的に	コードスタイルの一貫性が劇的に向上
★★★★☆	主要コマンド一覧	npm run dev / build / test / lint、docker compose up などよく使うもの	Claudeが自分でターミナル操作するときに正しいコマンドを使う
★★★☆☆	ワークフロー・ルール	TDD必須、Plan→Code→Testの順、コミットはatomicに、など	開発哲学を強制できる（特にTDD派・クリーンコード派に効果大）
★★★☆☆	禁止事項・ガードレール	絶対にしないこと（例: console.log残すな、秘密鍵をコードに書くな、など）	痛いミスを未然に防ぐ
★★☆☆☆	ドメイン知識・用語集	ビジネス用語、略語、特殊ルール（例: 「ユーザー」とは有料会員のみを指す）	ビジネスロジックでズレにくい
★★☆☆☆	よくあるバグ・注意点	これまで何度も間違えたポイント（例: timezoneは常にUTCで扱う）	同じ失敗を繰り返さなくなる

CLAUDE.mdの育て方

1. プロジェクトルートで最初に /init を実行

→ Claude Codeが自動でプロジェクトをスキャンして「とりあえず使える[CLAUDE.md](#)」を生成

2. 生成されたファイルをすぐ開いて人間が修正

- 自動生成は「骨組み」程度。必ず自分で肉付けする。
- 特に「プロジェクト概要」「技術スタック」「コーディング規約」は自分で書いた方が精度が高い。
 - a. ここで社内のガイドラインを参照したり、組み込むとよい

3. 小さく始めて、徐々に育てる（最重要）

- 最初は5～15行程度でOK。
- Claudeが間違った行動・コードを出したときに「これを防ぐためにCLAUDE.mdに追記」
 - これを繰り返すことで、自然に「そのプロジェクトに最適化されたCLAUDE.md」が育つ。

CLAUDE.mdの育て方

4. 肥大化させないための鉄則

1ファイルで200～300行を超えたら分割

→ .claude/rules/ フォルダに細かいルールを個別ファイル化（最近のアップデートで対応強化）

詳細なドキュメントは別ファイル（architecture.md など）にして、CLAUDE.mdからは「パス＋参照」するように記述する

主要ドキュメント

- 01_配布資料/03_CLAUDE.md_グローバルルール版.md - 組織全体で使用するグローバルルール
- 04_テンプレート/02_CLAUDE_template.md - プロジェクト固有のCLAUDE.md作成用テンプレート
- 04_テンプレート/01_要件定義ひな型テンプレート.md - 要件定義テンプレート
- 02_ハンズオン資料/02_ハンズオン手順書_第1回.md - ステップバイステップのハンズオン手順

5. チームで共有する場合の運用 ./CLAUDE.md ... チーム全員で共有するルール（Git管理）

./CLAUDE.local.md または ~/.claude/CLAUDE.md ... 個人設定（gitignore推奨）

CLAUDE.md

チートシート

CLAUDE.md に書くべきこと チートシート

AIに効くプロジェクト記憶ファイルの構成・書き方・調整 / 2026-05 時点 公式仕様準拠



CLAUDE.md とは / 階層

毎セッション先頭で全文ロード Claude が会話開始時に読むmarkdownファイル。システムプロンプトではなくユーザーメッセージとして注入されるため、強制ではなく文脈として書く。異体・無関係な履歴がユーザー Claude/CLAUDE.md → プロジェクト/CLAUDE.md または Claude/CLAUDE.md → /CLAUDE.local.md へ追加。ロード時は読み取り専用 managed policy → ユーザー Claude/CLAUDE.md → プロジェクト/CLAUDE.md または /CLAUDE.local.md へ追加。後で読み取り専用が更新されて優先される。

~/Claude/CLAUDE.md → ~/CLAUDE.md → ~/CLAUDE.local.md
ディレクトリは起動時、手は遅延 作業ディレクトリから上へ追った親の CLAUDE.md は起動時に全文ロード。サブディレクトリ配下はそのフォルダのファイルが Claude が読んだ時点で初めて読み込む。

managed policy は除外不可 macOS/Library/Application Support/ClaudeCode/CLAUDE.md 等の組織配布版は個人設定で除外できない。会社コーディング規約・セキュリティ方針を全員に強制する用途。

強制しない5hookを併用 CLAUDE.md はhookで書かれて保証ではない。コミット前 lint等の必須記述はPreToolUse/Stop hookやpermissions.denyで技術的に固める。

必ず書く中身

★ プロジェクト概要・構成 アーキテクチャ上の決定とディレクトリ配置。コードを読めば分かる説明ではなく、API/UIの置き場所など推測しづらい固定情報。
API handlers live in 'src/api/handlers/'

★ ビルド/テスト/実行コマンド Claudeが推測できないコマンド。標準のnpm test 以外に、単体テスト実行や開発チェックの作法を明記。
Run 'npm test' before committing

★ コードスタイル規約 言語デフォルトと異なるルールだけ。Claudeが既に知る一般慣習を書かない。
Use ES modules (import/export), not CommonJS

★ リポジトリ構造 プラットフォーム、PR規約、コミットメッセージ様式など、コードからは読めないチーム合意。

★ 環境の書・必須env var ローカルRedisがある。特定の環境変数が必要といった開発環境固有の情報。

★ 既知の落とし穴 非自明な挙動やよくハマる点。コードレビューでClaudeが知っているべきだった指摘は、その場でここへ追記する。

コマンド系

テストランナーと実行範囲 好みのテスト方法を指定。全件実行は避けるので単体優先など。Claudeが読み取り専用を要する。
Prefer running single tests, not the whole suite, for performance

完了時の検証手順 一連の変更後に型チェックやビルドを走らせる指示。pass/failの信号を返せる検証を持たせるを無人実行が閉じる。
Be sure to typecheck when you're done making code changes

CLIツールの利用指針 gh/awsl/gcloud等を使うよう明記。ghがあればPR作成やissue管理も文脈効果よくこなす。未利用ツールはhelpから学ぶ。

詳細APIドキュメントは書かない 長いAPI仕様は本文に貼るリンクで誘導。頻繁に変わる情報・チュートリアルは独立したファイル。

規約・作法

差分のあるスタイルだけ デフォルトと違う規約に絞る。「きれいに書く」のような自明な指示は逆効果。検証可能な具体で書く。
Use 2-space indentation

ブランチ/PR/コミット規約 命名規則やマージ方針などリポジトリ作法。チーム共有のため。CLAUDE.mdに書き加へコミット。

矛盾を放置しない 複数のCLAUDE.md・サブディレクトリ・Claude/rules/間で食い違ってもClaudeが任意に一方を選ぶ。定期的に確認して衝突を消す。

個人設定はCLAUDE.local.md サンドボックスURLや好みのテストデータなど共有不要の項目は、CLAUDE.local.mdにだけgitignoreへ。

@importと構成

@path で外部ファイル参照 @相対パス または @絶対パスで他ファイルを取り込む。相対パスはimport元基準で解決。最大4ステップまで再帰可能。
git workflow @docs/git-instructions.md

importしても文脈は減らない 取り込み済み起動時に全文ロードされ文脈を消費する。分別は整理目的であってトークン削減にはならない。

Source (公式・一次情報/リソース一覧)

- 公式メモリ仕様 code.claude.com/docs/en/memory
- 公式ベストプラクティス code.claude.com/docs/en/best-practices
- Skills code.claude.com/docs/en/skills

AGENTS.md@importで結合 ClaudeはAGENTS.mdを直接読まない。CLAUDE.md 冒頭に@AGENTS.mdと書き、その下にCLAUDE固有指示を足す両ツールが同一指示を読む。

@AGENTS.md/@path Claude Code/ide plan node for changes under 'src/billing/'。大規模はClaude/rules/へ 参照別にrulesファイルへ分割。paths.frontmatterで該当ファイルを開いた時だけロードする条件付ルールの書き、文脈を節約。
---logpaths:\n - "src/api/"/"*/.ts"\n---

HTMLコメントは保存メモ専用 <!-->でブロックコメントは文脈注入前に除去される。トークンを使わず人間向け記法を残せる。コードブロック内のコメントは保持。

書き方のコツ(階層)

200行未満を自己 17ファイル200行を超える 文脈消費が増え遅延が落ちる。各行に「これを消すとClaudeがミスするか」を問い、ノーなら削る。
if not needed -> cut it

強調語で追従率を上げる 守らせたい一行にIMPORTANTやYOU MUSTを付けて重みを足す。プロンプトと同じ感覚でチュニングする。

見出しと箇条書きで構造化 markdownヘッダーとbulletで関連指示をグルーピング。節の最初よりスキャンしやすく追従しやすい。

コードブロックに適用 挙動が動いたらレビューし定期的に否定。変更後はClaudeの挙動が実際に変わったかで効果を検証する。

曖昧さを減らす CLAUDE.mdに書いたのに質問が来る・ルールが無視されるなら、文言が曖昧か全体が長すぎてルールが埋もれている。

アンチパターン

肥大化で指示が埋もれる 長すぎるCLAUDE.mdは半分無視される。ルールが効かない時はまず指示を短く、強欲なく否定するかhookへ変換。

READMEの丸写し コードを読めば分かること・ファイル単位の説明は書かない。概要が必要なら@README.mdで参照させる。

頻繁に変わる情報の放置 詳細API仕様や移ろう手順を本文に書くこと古い情報が残り誤動作の元。リンク誘導かskillに任せろ。

都度しか要らない知識を重複 たまにしか使わないメイン知識やワークフローはskillへ。全話を書けず必要時だけロードされる。

自明な精神論 「clean codeを書け」のような中身の無い指示はノイズ。具体的・検証可能な記述に書き換える。

運用コマンド

操作	コマンド / 記法	効果
階層自動生成	/init	コードベースを解析しビルド/テスト/規約入り CLAUDE.mdを生成。既存があれば改善提案
メモリ編集	/memory	ロード中のCLAUDE.md・CLAUDE.local.md・rulesを一覧。読み込みに失敗したら、update memoryのディレクトリ
会話から追記	#テキスト	プロンプト冒頭のを、その場の内容をメモリへ保存。「CLAUDE.mdに追加して」と頼んでもよい
別ファイル追加	@path/ho/ file	@相対/絶対パスで起動時に全文読み込み。最大4ステップ再帰
AGENTS.md統合	@AGENTS.md	CLAUDE.md冒頭に書くこと既存AGENTS.mdを読み込み、下にCLAUDE固有指示を追加できる
話題別ルール	./Claude/rules/*.md	話題ごとに分割。paths.frontmatterで該当ファイル時のみロードする条件付きにできる
不要なものを除外	./Claude/Exclude	settings.local.jsonでglob指定し、無関係な優先 CLAUDE.mdを読み込みない。managed policyは除外不可

Tips・補足

守らせたい一行にはIMPORTANT/YOU MUSTを付けて追従率を上げる
サブディレクトリのCLAUDE.mdはそのフォルダのファイルを読んだ時に遅延ロード。monorepoの局所限的に有効

ClaudeはAGENTS.mdを直接読まない。@AGENTS.md importがsymlinkで両ツールの指示を一本化する

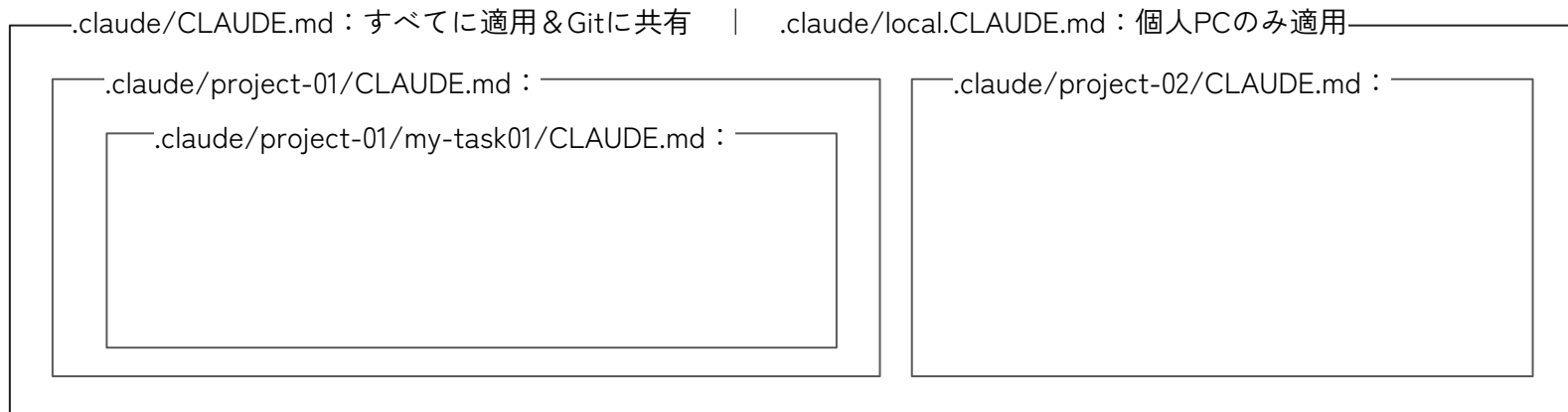
auto memory(MEMORY.md)はClaudeが自動で書く学習メモ。先頭200行/25KBのみ有効。CLAUDE.mdとは別系統

強制が必要な記述はCLAUDE.mdではなくhookやpermissions.denyで技術的に固める

Copyright © 2026 Givery, Inc. All Rights Reserved.

- Hooksガイド code.claude.com/docs/en/hooks-guide
- Settings code.claude.com/docs/en/settings
- context window code.claude.com/docs/en/context-window

- [CLAUDE.md](#)はレイヤーに合わせて2-3層で作成・設定する
- [CLAUDE.local.md](#) という名前にすると個人でのみ使用可能
- 1ファイルにつき200-300行がMAX。書きすぎは厳禁
- @メンションで別で書いてあるファイルを参照させることも可能



**「これを消したら
ミスが起きるか？」**

CLAUDEに含める・含めないもの

含める	含めない
Claudeが推測できないBashコマンド	コードを読めばわかること
デフォルトと異なるコードスタイル	言語の標準慣習
テスト実行方法	詳細なAPIドキュメント（リンクで）
リポジトリのルール（ブランチ命名等）	頻繁に変わる情報
アーキテクチャ上の決定事項	ファイル毎の説明
環境変数などの開発環境の癖	「きれいなコードを書け」等の自明な指示

新規プロジェクトにおける CLAUDE.mdの定義と育て方

プロジェクト管理と 標準開発フロー

リサーチ×md1化ハンズオン

スライ ド Skill

配布した『gen-slide-template』について

3つの別のアプローチから、
スライドのテンプレートに沿ってAIによるスライド生成を検証しました。

その方法をClaudeCodeで呼び出せるようにしたKitです。

ただし、AI80%-修正10%-人間による調整10%程度

①adl-html-kit

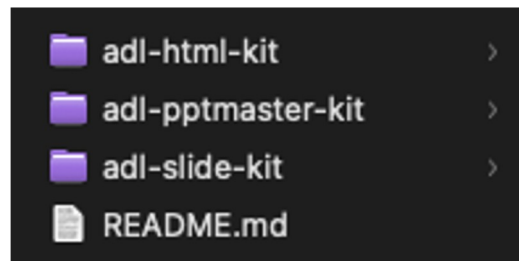
→AIが単一のHTML形式のスライド（1920/1080）を生成する

②adl-pptmaster-kit

→AIが1ページずつのスライドをSVGで描画し、PPTXに変換

③adl-slide-kit

→スライドの内容をYAMLで記述し、Pythonで調整、PPTXを作成する



配布した『gen-slide-template』について

3つの別のアプローチから、
スライドのテンプレートに沿ってAIによるスライド生成を検証しました。

その方法をClaudeCodeで呼び出せるようにしたKitです。

ただし、AI80%-修正10%-人間による調整10%程度

①adl-html-kit

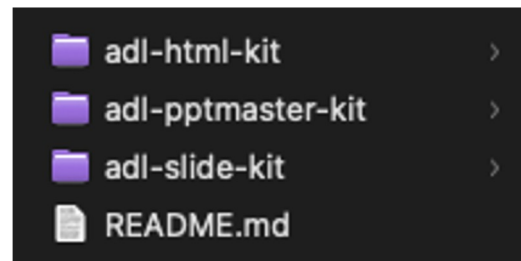
→AIが単一のHTML形式のスライド（1920/1080）を生成する

②adl-pptmaster-kit

→AIが1ページずつのスライドをSVGで描画し、PPTXに変換

③adl-slide-kit

→スライドの内容をYAMLで記述し、Pythonで調整、PPTXを作成する





株式会社ギブリー (Givery, Inc.)
〒150-0036
東京都渋谷区南平台町15-13帝都渋谷ビル8F
TEL 03-5489-7188 / FAX 03-5489-7189

本資料の二次転載禁止について

本資料に掲載されている情報は、著作権および知的財産権により保護されています。

本資料の内容を許可なく、二次転載（複製、配布、公開、改変、引用、商用利用等）することを固く禁じます。

違反した場合、著作権法および関連法に基づく法的責任が発生することがありますので、ご注意ください。

ご理解とご協力のほど、よろしくお願いいたします。

