E1 Claude Code

はじめてのAIエージェント：フォルダの置き場所・分け方・管理

先にここで、手を動かす。置き場所（クラウド同期は避ける）、顧客・アプリ・案件の分け方、AIに頼んで自分の事業に合わせる方法、指示書・現在地・決定ログ、完成の基準、会話の区切り方、戻せる仕組み、最初の30分

更新日：2026-10-03（2026年10月時点の情報）｜HUMAC 伊藤晴通

対象ツール：Claude Code

この資料は、**Claude Code**（Anthropic のAIエージェント。Claudeのデスクトップアプリの「Code」タブ）を前提に書いています。ほかのAIエージェント（OpenAI の Codex、Google のエージェントなど）は、**画面・用語・ファイル名・設定が異なります**。

対象の方：開発者ではない方（経営者・事務・現場の担当者）。プログラミングは不要です。**コードは書かず、AIエージェントに頼んで、自分の業務アプリをつくる**ところまでを、順に進めます（ターミナルの黒い画面は、使わない形で書いています）。

AIエージェントを使う前に、まず「場所」を決める

AIエージェントは、開いたフォルダの中を「自分の職場」として働く。資料があちこちに散らばっていると、AIも人も迷う。最初にやるのは、頼み方の工夫ではなく**「場所づくり」。1つの仕事＝1つのフォルダ**にして、その中に「指示書」「現在地」「決めたこと」を置くと、別の日・別の人・別のPCでも、同じ品質で続きができる。

E1とE2は、セットで使う：先にここ（E1）で、手を動かす

このページ（E1）は「**置き場所と進め方**（どこに・どうやって）」。**先に、ここで、手を動かして、フォルダを作る**。続けてE2 ハーネスエンジニアリング入門（**考え方と6つの部品**：なぜ・何を）を読むと、「なぜ、そうするのか」が、腑に落ちる。勉強会・サロンでは、**E1（手を動かす）→ E2（考え方）**の順に使う。

1. 先に知っておく、2つの制約

このページのルールは、すべて次の2つから来ている（Claude Code の公式ドキュメントと、Anthropic の技術解説による）。

① AIは、毎回「白紙」から始まる

新しく会話を始めるたびに、AIは前回のことを覚えていない。**前回の続きを引き継ぐ唯一の方法は、ファイルに書いておくこと**。だから「指示書」「現在地」「決定ログ」が要る。

② 作業机（文脈）は小さく、いっぱいになるほど鈍る

会話・読んだ資料・実行結果は、すべて「作業机」に載る。机がいっぱいになると、**前の指示を忘れたり、間違いが増える**（「context rot」と呼ばれる）。だから「整理して、小さく保つ」「話題ごとに区切る」ことが、最も大事な習慣になる。

2. どこに置くか：クラウド同期フォルダは、避ける

仕事のフォルダ（AIに開かせるフォルダ）は、**パソコンの中（ローカル）**に置く。Google ドライブ・iCloud Drive・OneDrive・Dropbox のような**クラウド同期のフォルダの中には、置かない**のが、おすすめ。

置き場所	作業フォルダ	ひとこと
パソコンの中 **Mac：~/Projects** **Windows：C:\Users\名前\Projects**	◎	ホーム（ユーザー）フォルダの直下に、Projects を作る。「デスクトップ」「書類」の中には、作らない（下の注意）
Google ドライブ（Drive for desktop）	×	資料の保管・共有には、向いている
iCloud Drive	×	Macの「デスクトップと書類フォルダ」の同期がオンだと、デスクトップ・書類の中身が iCloud Drive に保存される（Apple 公式）
OneDrive	×	「フォルダーのバックアップ」をオンにしていると、デスクトップ・ドキュメントなどが、同期の対象になる（設定による）
Dropbox など	×	同じ理由

なぜ、避けるのか（5つの理由）

理由	起きること
① Gitの管理ファイルが壊れる	同期サービスが、同時に変わったファイルを、「main 2」のような**番号つきの複製**にしてしまい、`fatal: bad object` というエラーで、保存・送信ができなくなる報告がある（iCloud Drive での事例）。直すのは、難しい

理由	起きること
② 小さなファイルが大量で、同期が止まる	Gitや、AIが作る一時ファイルは、数千個になる。OneDrive が応答しなくなった報告があり、**OpenAI の Codex でも、「クラウド同期フォルダでの作業に、警告を出してほしい」という課題**が記録されている
③ 「オンラインのみ」のファイル	OneDrive の「オンラインのみ」のファイルは、**開くたびにダウンロード**される（Microsoft 公式）。Google ドライブの「ストリーミング」も、ファイルは主にクラウドにあり、開いたときにダウンロードされる（Google 公式）。AIが、大量のファイルを読むと、遅くなる・失敗する
④ 鍵や実データが、自動でクラウドに上がる	知らないうちに、クラウドにコピーされる（D1・E6）
⑤ 同時に書き換えて、衝突する	AIが、多数のファイルを、短時間に書き換える。同期と衝突し、複製ファイルが増える

「必ず壊れる」わけではない。個人のメモ用途で、問題なく使っている人もいる。ただし、壊れたときに直すのは難しく、非開発者には、**避けるのが安全**。

最善の提案：3つの役割に分ける

1

① 作業場所＝パソコンの中

Projects（ホーム直下）。AIが開く・Gitがある場所

2

② バックアップ＝Git → GitHub（非公開）

仕事の履歴と、控え（E4）。加えて、パソコン全体のバックアップ

3

③ 受け渡し・共有・保管＝クラウドストレージ

完成した資料の共有、会社の資料室（Google 共有ドライブ。W2）

やりたいこと	やり方
クラウドにある資料を、AIに読ませたい	必要なものを、**ローカルの inputs/ にコピー**してから読ませる（コピーも、AIに頼める）
できた成果物を、お客様・社内に共有したい	完成版を、**クラウドの共有フォルダに、コピー**する（仕事のフォルダは、そのまま、ローカルに置く）
複数のパソコンで、同じ仕事を続けたい	クラウド同期ではなく、**GitHubで同期**する（保存して送る／取り込む。E4）
会社として、資料をAIの根拠にしたい	**共有ドライブを資料室**にして、Gemini Notebook に入れる（W2）。ここは、同期ではなく、資料の置き場所

どうしても、クラウドの中で作業するなら（最低限の条件）

- **Git を使わない、資料中心のフォルダ**に限る（`.git` を、同期の中に置かない）。
- Google ドライブは、「ストリーミング」ではなく、**「ミラーリング」**（ファイルを、常に、パソコンとクラウドの両方に持つ）にする。OneDrive は、そのフォルダを「常にこのデバイスに保持」にする。
- AIに、大量のファイルを作らせない。鍵・実データを置かない。

自分のフォルダが、同期の中にないかを、AIに確認してもらう（コピーして使う）：

コピーして使う

いま選んでいるフォルダの場所（パス）を調べて、iCloud Drive・OneDrive・Google ドライブ・Dropbox などの「同期フォルダ」の中ではないか、確認してください。中だった場合は、ホーム（ユーザー）フォルダ直下の Projects への、移動の手順を提案してください。移動・削除は、私の承認を得てからにしてください。

講師の実例

作業フォルダ（Projects）は、パソコンの中に置き、**iCloud の同期の対象外**にしている。バックアップは、GitHub の非公開の金庫。お客様の書類の保管・共有は、Google ドライブ。**「作業場所」と「保管・共有」を分ける**のが、基本の考え方。

3. 何の単位で分けるか：顧客ごと？ アプリごと？ 案件ごと？

「1仕事＝1フォルダ」の、「仕事」を、どう切るか。**正解は、事業ごとに違う**。ここでは、決めるための「基準」と「型」を示す。そのうえで、**自分の事業に合わせて決める**（AIと一緒に決める方法は、後半）。

切り方は、4つ

切り方	向く仕事	メリット	注意
顧客ごと	継続して関わる（顧問・保守・士業・コンサル）	顧客共通のルール・用語・担当者・守秘を、1か所に書ける	案件が混ざる。**顧客の中を、案件で、さらに分ける**
案件ごと	期限と成果物がある（見積〜納品、申請、イベント）	終わったら、まるごと退避（`_archive`）できる	同じ顧客の案件が、散らばらないように、顧客の下にまとめる

切り方	向く仕事	メリット	注意
アプリごと	つくって、育てていく（社内アプリ、顧客向けアプリ）	**Git の単位＝アプリ**。履歴・共有・引き継ぎが、アプリごとにきれい	顧客が複数のアプリを持つなら、顧客の下に、アプリを置く
業務ごと	自社の仕事を良くする（月次の請求チェック、問い合わせ返信など）	「F1 の棚卸し」の単位と、そろう	部署が違うと、見せてよい範囲が違う。部門で束ねる

迷ったら：5つの質問（上から順に）

#	質問	答えの使い方
1	**AIに見せてよい範囲が、同じか？**（守秘の境界。D1）	**最優先**。別の顧客・別の部署・信号の色が違う情報は、**同じフォルダに入れない**（AIが取り違える・漏らす）
2	AIに渡す、資料・ルール・言葉が、同じか？	同じなら、同じフォルダ。違うなら、分ける
3	引き継ぐ相手（担当者）が、同じか？	担当が変わる単位で、切ると、引き継ぎが楽
4	終わりがあるか？	案件＝あり（終わったら退避）。アプリ・業務＝なし（育てる）
5	契約・請求の単位は？	顧客か案件の単位と、そろえると、あとで整理しやすい

指示書は、3層に重ねられる（親のフォルダの指示書も、効く）

Claude Code は、作業フォルダの CLAUDE.md だけでなく、**その上のフォルダの CLAUDE.md も、連結して読む**（公式）。これを使うと、「顧客共通のルール」を、1回だけ書ける。

フォルダの形

📁 Projects
📁 clients … 顧客ごと
📁 tanaka-sekkei … 顧客A
CLAUDE.md … ②顧客共通（守秘・用語・担当者・連絡の作法）
📁 2026-monthly-check … 案件・業務 ◀ AIを開くのは、ここ
CLAUDE.md ／ STATUS.md ／ docs/ … ③この案件だけのルールと現在地
📁 2026-booking-app … 同じ顧客の、別の案件
📁 apps … 自社のアプリ（1アプリ＝1フォルダ）
📁 work … 自社の業務（月次の請求チェックなど）
（全体の指示書 ~/.claude/CLAUDE.md … ①全部の仕事に共通）

- **AIに開かせるのは、いちばん下（案件・アプリ・業務）のフォルダ**。顧客フォルダを開くと、同じ顧客の別の案件の資料も、AIの範囲に入る。
- 顧客共通のルールは、顧客フォルダの CLAUDE.md に1回だけ書き、各案件には、案件だけの内容を書く。

事業のタイプ別：型の例

事業のタイプ	おすすめの切り方	フォルダの例
顧問・士業・コンサル（顧客に継続して関わる）	**顧客 → 案件**。顧客共通の指示書＋案件ごとの指示書	`clients/顧客略称/2026-案件名/`
受託でアプリをつくる	**顧客 → アプリ**。Git の単位は、アプリ	`clients/顧客略称/アプリ名/`
自社のアプリを育てる	**アプリごと**	`apps/アプリ名/`
店舗・製造・事務（自社の業務を良くする）	**業務ごと**。部署で束ねる	`work/経理/月次請求チェック/`
セミナー・発信・学習	**テーマごと／開催ごと**	`seminar/2026-10-17-主催名/`

Git・GitHub との関係

- **1つのフォルダ（Gitの単位）＝共有・引き継ぎの単位**。顧客Aと顧客Bを、同じ保存先（リポジトリ）に入れない。
- お客様と共有する場合は、**案件・アプリの単位**で、共有する（顧客フォルダ全体を共有しない）。

名前の付け方

- 顧客：**半角の英小文字とハイフン**の略称（例：tanaka-sekkei）。会社名の変更があっても、フォルダ名は、変えない。
- 案件：**西暦-内容**（例：2026-monthly-check）。アプリ：アプリの名前。
- 日本語の説明は、フォルダの中の文書（**運用ルール**.md など）に書く。

自分の事業に合わせて、決める手順

1

1 仕事を書き出す
思いつく順に、10個。F1のステップ1

2

2 まとめる
「同じ資料・同じルール・同じ相手」で、グループにする

3

3 線を引く
AIに見せてよい範囲（守秘）が違うところで、必ず分ける

4

4 名前の規則を決める
顧客・案件・アプリの付け方

5

5 3か月使って、見直す
はじめから、完璧にしない。変えるのは、普通

AIに頼んで、決める（コピーして使う）

一般論ではなく、**自分の事業の言葉**で、AIに質問してもらいながら決める。

コピーして使う

私の事業は、〔業種・人数・何をしているか〕です。
仕事の例：〔月次の請求チェック／顧客Aの保守／新しいアプリの開発 など、思いつく限り〕
AIに見せてはいけない情報：〔個人情報・決算書・他のお客様の情報 など〕
仕事用のフォルダの構成案を、3つ出してください。それぞれ、①フォルダの例（図）、②向く理由、③取り違え・情報漏れのリスク、④3か月後に困りそうなこと、を比べてください。
いきなり決めずに、先に私に質問してください（1回3問まで）。フォルダを作る前に、必ず、私の承認を得てください。

いまのフォルダが散らかっているとき（提案だけ。移動はしない）

コピーして使う

〔いまのフォルダ〕の中身を調べて、仕事ごと・顧客ごとの整理案を出してください。ファイルの移動・削除・名前の変更は、しないでください（提案だけ）。案には、①新しいフォルダの構成、②どのファイルを、どこへ移すか（一覧）、③迷うもの（私に確認が必要なもの）、④消さずに退避するもの（_archive）を、入れてください。

顧客共通の指示書を作る

コピーして使う

顧客〔略称〕のフォルダに置く、顧客共通の CLAUDE.md を作りたいです。いきなり書かずに、先に私に質問してください（守秘のルール、用語、担当者、連絡の作法、やってはいけないこと、使う資料の置き場所など）。聞き終えたら、200行以内で、箇条書きで書いてください。個別の案件の内容は、書かないでください。

見直しのサイン

AIが、**別の案件・別のお客様の話を、持ち出す**／同じ説明を、何度もしている／探しものが多い／1つのフォルダで、共有する相手が違う。→ 分け方を、見直すタイミング。フォルダの移動は、**Git の履歴が切れる**ので、早めに。

4. 基本の構成：1仕事＝1フォルダ

「仕事のフォルダ」は、1か所にまとめる（デスクトップや書類フォルダにばらまかない）。名前は半角の英小文字とハイフンにすると、扱いやすい。

最初はこれだけ（軽量版）

フォルダの形

- 📁 Projects … 仕事のフォルダは、すべてここ
  - 📁 monthly-invoice-check … 1つの仕事＝1フォルダ
    - CLAUDE.md … ①指示書（AIが最初に読む。薄く）
    - STATUS.md … ⑤現在地（今どこまで・次にやること）
    - 📁 docs
      - 決定ログ.md … ⑤決めたことと理由
    - 📁 inputs … 材料（請求書PDFなど。AIは読むだけ）
    - 📁 outputs … AIが作ったもの（下書き・集計表）
    - 📁 _archive … 退避（消さずに、ここへ移す）

慣れたら足す（フル版）

フォルダの形

📁 monthly-invoice-check
CLAUDE.md ／ STATUS.md
RAID.md … ⑥気がかり帳（リスク・課題・前提・依存）
📁 docs
運用ルール.md … この仕事の真実源（詳しい決まり）
決定ログ.md ／ 用語集.md
📁 .claude
📁 skills … ②スキル（この仕事の手順書）
settings.json … ③権限・④関所の設定
📁 data … 実データ（⚠ 共有・保存の対象から外す）
.gitignore … 鍵（.env）・実データを、保存の対象から外す

AIは「開いたフォルダ」の中で働く

これは便利さであり、安全装置でもある。**フォルダの外は、AIに触らせない**（権限の範囲）。別の仕事の資料が同じフォルダにあると、AIがそちらを参照して取り違える。2つの仕事を、1つのフォルダで進めない。

5. 各ファイルの役割（E2の6部品との対応）

E2の部品	置き場所	役割	書き方のコツ	更新のタイミング
① 指示書	`CLAUDE.md`	AIが毎回最初に読む。この仕事の目的、置き場所、禁止事項	短く。目安は200行以内。「消してもAIが間違えない行」は消す（E2の4）	同じ訂正を2度したとき
② スキル	`.claude/skills/`	この仕事の手順書。呼んだときだけ読まれる	手順は指示書ではなく、スキルへ（F1の手順書）	手順が固まったとき

E2の部品	置き場所	役割	書き方のコツ	更新のタイミング
③ 権限	.claude/settings.json、フォルダの範囲、.gitignore	どこを読み、どこに書き、何を実行してよいか	最初は確認あり。安全と分かった操作だけ許可リストへ	新しい道具を使うとき
④ 関所	承認の設定、outputs/、_archive/	送信・公開・削除・支払いの前に、人が確認する	「書き込み先は outputs だけ」「消さずに _archive へ」と決めておく	外に出る操作を足すとき
⑤ 記憶	STATUS.md、docs/決定ログ.md	現在地と、決めたことの理由。次の会話に持ち越す	STATUS は「済んだこと／次の一手／待っていること」だけ。決定ログは、日付・決めたこと・理由・却下した案	毎回の作業の最後
⑥ 点検	RAID.md、週1回の点検	ルールと実態のずれ、残っている課題を確かめる	R（リスク）A（前提）I（課題）D（依存）で分け、閉じたら消す	週1回

6. 最初の30分：手を動かして、フォルダを作る

勉強会・サロンでは、ここを実際にやる。まず、小さくて、失敗しても取り返しがつく仕事を1つ選ぶ（例：毎月の請求書チェック、問い合わせメールの下書き）。

1

1 フォルダを作る
Projects の中に、仕事の名前（英小文字とハイフン）で。inputs・outputs・_archive も作る

2

2 AIに、質問させる
いきなり指示書を書かない。AIに「聞き取って」もらい、答えながら作る

3

3 指示書と現在地を作る
CLAUDE.md と STATUS.md を、AIに書かせる。人が読んで直す

4

4 関所を決める
送信・公開・削除は、人が押す。書き込み先は outputs だけ

5

5 小さな仕事で試す
1件だけ頼む。結果を人が確認し、直すところを指示書へ

6

6 保存して終える
STATUS に今日の到達点を書き、変更を保存する（Git の準備は E4）

手順2：AIに「聞き取って」もらう（コピーして使う）

コピーして使う

この仕事（〔仕事の名前〕）の指示書（CLAUDE.md）を作りたい。いきなり書かずに、まず私に質問してください。
聞いてほしいこと：目的、出来上がりの形、使う資料と置き場所、やってはいけないこと、言葉づかい、迷ったときの判断。
1回に聞くのは3問まで。聞き終えたら、200行以内で、箇条書きで CLAUDE.md を書いてください。書いたら、足りないところを私に確認してください。

手順3：現在地ファイルを作る

コピーして使う

STATUS.md を作ってください。見出しは「済んだこと」「次の一手」「待っていること」「決めたこと（日付つき）」の4つだけ。今日の作業を反映して、次に私が開いたときに、これだけ読めば続きができる状態にしてください。

終わるときの合言葉（毎回）

コピーして使う

今日の作業を締めます。STATUS.md を更新し、決めたことと理由を docs/決定ログ.md に追記して、変更を保存してください。次回の最初にやることを、STATUS の先頭に書いてください。

「変更を保存してください」は、Git（E4）の準備ができていれば、そのまま動く。まだなら、この一文を除いて使う。

7. 仕事の進め方の型

公式ドキュメントと、長時間働くエージェントの設計（Anthropicの技術解説）に共通する、うまくいく進め方。

コマンドの名前は、覚えなくてよい

この資料は、**Claudeのデスクトップアプリ（Codeタブ）の画面のボタンと、Claudeへの頼み方**だけで進められるように書いてある。ターミナル（黒い画面）は開かなくてよい。コマンドは、慣れた人向けの「参考」として、小さく添えるだけにする。

① 先に調べて、計画してから、実行する

1. **探索**
   読むだけ。何も変更しない（プランモード）
2. **計画**
   何をどの順で行うか、案を出させる。人が直す
3. **実行**
   計画に沿って進める
4. **確認・保存**
   結果を確かめて、変更を保存する

いきなり実行させると、「間違った問題を解く」ことがある。ただし、**1文で説明できるほど小さな変更**（誤字の修正など）は、計画を飛ばしてよい。

② 「完成の基準」を先に渡す

AIは「できたように見えた」ところで止まる。**確かめる方法が無いと、人が毎回、間違いを見つける係になる**。次のように、確認の手段を一緒に渡す。

曖昧な頼み方	完成の基準を付けた頼み方
請求書をチェックして	請求書PDF 30件を読み、金額・税率・日付を表にして。取引先リストと一致しないものを一覧にして。**最後に、件数の合計が合っているか確かめて、結果を見せて**
メールを書いて	お客様への見積案内メールを下書きして。条件：です・ます調、200字以内、金額は見積書の数字をそのまま写す。**書いたあと、条件を1つずつ満たしているか確認して**

- 「できました」という言葉ではなく、**確認した結果（数字・実行結果・画面）を見せて**、と頼む。証拠を見るほうが速い。
- 作った本人（AI）が、そのまま採点しない。**別の会話・別のAIに、「抜けがないか」を見てもらう**と、思い込みが入りにくい（作る人と見る人を分ける）。ただし、見る係に「問題を探して」と頼むと、必ず何か指摘する。**正確さと要件に関わる指摘だけ**、と範囲を決める。

③ 1度に1つ。終わったら確認して、保存する

長い仕事を一気に頼むと、AIは「全部できた」と早く宣言したり、途中で作業机がいっぱいになって中途半端に終わる、という失敗が起きやすい。Anthropicの設計では、次の工夫で防いでいる。

起きること	対策
「全部できた」と早く宣言する	**やることの一覧**（チェックリスト）を最初に作る。人が確認してから、チェックを付ける
確認が足りないのに、完了にする	確認が終わったものだけ、完了にする

起きること	対策
前の作業の続きが分からなくなる	進捗のメモ（STATUS）と、変更の保存を、1つ終わるごとに残す
やり方が毎回変わる	起動や実行の手順を、1か所（スキルやスクリプト）にまとめる

④ 会話は区切る：1つの仕事＝1つのセッション

Claudeのデスクトップアプリでは、1つの会話を「セッション」と呼ぶ。左のサイドバーに一覧が並び、いつでも開き直せる。**仕事が変わったら、新しいセッションにする**（サイドバーの「＋ 新しいセッション」。MacはCmd＋N、WindowsはCtrl＋N）。

状況	やること
別の仕事の話に移る	新しいセッションを始める。関係ない話が混ざると、精度が落ちる
同じことを2回直しても直らない	新しいセッションで、学んだことを入れた、より具体的な頼み方で、やり直す。直し続けるより、早い
会話が長くなった・今日は終わりにする	**「終わりの合言葉」**（6章）で、STATUS.md と決定ログに書いてもらう。次は、新しいセッションで、STATUS を読ませて始める
続きを別の日にやる	サイドバーのセッション一覧から開き直す。または、新しいセッションで、STATUS.md を読ませる。セッションの名前は、上のタイトルをクリックして、分かりやすく付け直せる
調べものが大量になる	調べる役を「別の担当（サブエージェント）」に任せるよう頼む。結果の要約だけが戻るので、作業机が汚れない

講師のやり方：「会話を整理する」より「ファイルに書いて、新しく始める」

講師自身は、会話の要約（`/compact`）や、会話の消去（`/clear`）のコマンドは、ほとんど使っていない。作業の最後に、**「締めます」の一言で、STATUS・決定ログ・保存まで書き出し**、次は新しいセッションで、続きから始める。**覚えておくことは、会話の中ではなく、ファイルの中に置く**。これなら、コマンドを覚えなくても、毎回、同じ品質で続けられる。

参考（慣れた人向け）：会話を要約して縮める `/compact`、会話を消して新しく始める `/clear`、前の会話を選んで再開する `--resume`。ただし、要約すると、会話の中だけで伝えた指示は失われやすい。残したいことは、ファイルに書く。

⑤ いつでも戻せるようにする

- **変更は、確認してから受け入れる**：最初は「確認あり（Manual）」のモードにしておくと、Claudeがファイルを書き換える前に、**何がどう変わるか（差分）を画面で見て、「承認」か「却下」を選べる**。受け入れたあとも、変更のあった行数（例「＋12 −1」）をクリックすると、変更の一覧を、あとから見直せる。
- **取り消したいとき**：「さっきの変更を、取り消してください」と頼む。ただし、AIの巻き戻し機能で戻せるのは、**AIが直接編集したファイルだけ**。コマンドで消したものや、外部への操作は戻せない。バックアップの代わりにはならない。
- **変更履歴（Git）が本命**：フォルダごと履歴で管理すると、「昨日の状態に戻す」ができる。難しい操作は不要で、Claudeに「保存して」と頼めばよい。**導入と設定は E4 Gitの導入と設定**（ターミナルなしで進められる）。
- **消さずに退避する**：不要に見えるファイルも、`_archive` へ移すだけにする。消すのは、最後に人が決める（E2の事故の実例）。

⑥ 同じフォルダで、2つのセッションを同時に動かさない

同じファイルを、2つのセッションが同時に書き換えると、衝突する。並行して進めるなら、**作業用のコピー（ワークツリー）**か、別のフォルダに分ける。デスクトップアプリでは、新しいセッションを作るときに、ブランチ名の横の「worktree」を選ぶと、そのセッション専用の独立したコピーで動く（Gitが必要。→ E4）。初めは「1フォルダ・1セッション」で十分。

8. 安全な置き場所の設計（E2の「権限」「関所」とつなぐ）

置き場所	AIの権限	信号
inputs/	読むだけ	🟢 〜🟡 公開情報／匿名化した資料
outputs/	書いてよい（下書きまで）。送信・公開は人	🟡
_archive/	移すだけ。消さない	—
data/（実データ）	必要なときだけ、人が許可。保存・共有の対象から外す	🔴 個人情報・決算書など（D1）
鍵・パスワード	ファイルに書かない。専用の金庫（Vault）に置く（→ E6）。AIに、鍵のファイルを読ませない設定も入れる	🔴

- **「必ず守らせたい」ことは、指示書に書くだけでなく、仕組みで止める**：指示書はAIへの「お願い」で、強制ではない。権限の設定や、決まった場面で自動で動くチェック（フック）で止める（E2の4-1）。

- **最初は「確認あり（Manual）」で始める**：入力欄の横のモード選択で選べる。Claudeがファイルを書き換えたり、コマンドを実行したりする前に、確認が出る。慣れて、安全と分かった操作だけ、許可する範囲を広げる。確認ボタンを、読まずに押し続けるようになったら、設定を見直すサイン。

9. 名前と整理のルール

- **日付と版を名前に入れる**：「最新版（修正）2.docx」は、すぐ嘘になる（→ B2）。
- **日本語のファイル名もよい**：「運用ルール.md」「決定ログ.md」は、見た瞬間に中身が分かる。勉強会で説明なしに伝わる、という利点もある。
- **真実源は1か所**：同じ内容を2か所に書かない。指示書には「詳しくは docs/運用ルール.md」と案内だけを書く。
- **他のAIツールと共用するなら**：多くのAIツール（20以上、OpenAI の Codex など）は、「AGENTS.md」という共通の指示書を読む。Claude は、CLAUDE.md が無いときだけ AGENTS.md を読む。両方使うときは、CLAUDE.md の中から取り込む。

10. フォルダを「育てる」：週1回の点検

最初から完璧なフォルダは作れない。**失敗を1つずつ、ファイルに書き足して育てる**。

こんなとき	やること
同じ間違いを、2度された	指示書に1行足す（具体的に、守れたか確かめられる形で）
同じ説明を、また入力した	指示書か決定ログに書く
指示書が長くなってきた	手順はスキルへ、一部の場所だけのルールは近くのファイルへ（E2の4-5）
AIが、指示を守らなくなった	指示書が長すぎて埋もれていないか、指示が食い違っていないかを見る。1行だけ強調してもよい

コピーして使う

（週1回の点検用）
このフォルダを点検してください。①CLAUDE.md が200行以内か、古い指示・食い違いがないか ②STATUS.md が最新か ③RAID.md の閉じられる項目 ④_archive に移したほうがよいファイル ⑤鍵や実データが、保存の対象に入っていないか。結果を表で報告してください。直す前に、私の確認を取ってください。

11. よくある失敗と、直し方

何でも1つの会話でやる

別の話が混ざり、作業机が散らかる。話題ごとに会話を新しくする。

直し続ける

2回直して直らなければ、新しい会話で、具体的な頼み方からやり直す。

指示書が長すぎる

半分は読まれない。「消しても間違えない行」は消す。手順はスキルへ。

確認の手段が無い

「もっともらしい」結果をそのまま使ってしまう。完成の基準と、結果の証拠を求める。

範囲を決めずに調べさせる

大量の資料を読み、作業机がいっぱいになる。範囲を絞るか、別の担当に任せる。

混ぜる

2つの仕事を1つのフォルダで進めると、別の資料を参照して取り違える。1仕事1フォルダ。

鍵を書いてしまう

共有やバックアップで漏れる。鍵は金庫に置き、保存の対象から外す。

「最新版」だらけ

日付と版を名前に入れる。古い版は _archive へ。

12. はじめての日のチェックリスト

□	確認すること
□	仕事を1つ選んだ（小さく、失敗しても取り返しがつく）
□	1仕事＝1フォルダを作り、inputs・outputs・_archive を用意した
□	AIに質問させて、CLAUDE.md を作り、200行以内で、人が読み直した
□	STATUS.md を作った（済んだこと／次の一手／待っていること）
□	送信・公開・削除は人が押す、と決めた（書き込み先は outputs だけ）
□	鍵や実データを、フォルダに置かない、または保存の対象から外した
□	1件だけ試して、完成の基準で確かめた

□	確認すること
□	変更履歴（Git）の準備ができた（→ E4）。まだなら、次回までの宿題にする
□	終わるときに、STATUS を更新し、変更を保存した
□	次回の最初にやることを、STATUS の先頭に書いた

勉強会・サロンでの使い方（E1とE2をセットで）

1. **E1（60分）**：まず、置き場所（2章）と、分け方（3章）を、自分の事業で決める。そして、6章の「最初の30分」を、全員で実際にやる。各自の仕事を1つ選び、フォルダ・指示書・STATUS を作り、1件だけ試す。
2. **E2（30分）**：手を動かしたあとで、「なぜ、そうするのか」。ハーネスとは何か、6つの部品、mdファイルの役割と行数の限界、事故から生まれたルール。
3. **E4（事前または当日に別枠で）**：Git と GitHub の導入と設定（①②を実際にやる）。ターミナルは使わない。
4. **宿題**：1週間、終わりの合言葉で STATUS を更新する。翌回に、「2度された間違い」を指示書に足した例を持ち寄る。

関連資料

- E2 ハーネスエンジニアリング入門 … 考え方と6つの部品。mdファイルの行数の限界
- E4 Gitの導入と設定 … 変更履歴という「保存の金庫」。ターミナルなしで進められる
- F1 業務の棚卸しと仕組み化 … 手順書の書き方（スキルの材料）
- B2 データ管理の型 … 資料の置き方・名付け方
- D1 信号機ルール … 入れてよい情報・いけない情報
- G1 1日集中合宿 ／ G2 2日集中合宿

出典（2026年10月確認）

- Best practices for Claude Code（Anthropic 公式） (https://code.claude.com/docs/en/best-practices)：文脈の制約、完成の基準、探索→計画→実装、会話の区切り方、巻き戻し、並行作業、よくある失敗
- How Claude remembers your project（Anthropic 公式） (https://code.claude.com/docs/en/memory)：CLAUDE.md・自動メモリ・AGENTS.md の扱い
- Effective harnesses for long-running agents（Anthropic） (https://www.anthropic.com/engineering/effective-harnesses-for-long-running-agents)：進捗ファイル、機能リスト、変更の保存、1度に1つ、確認してから完了にする
- Effective context engineering for AI agents（Anthropic） (https://www.anthropic.com/engineering/effective-context-engineering-for-ai-agents)：文脈は有限の資源、外部ノート、サブエージェント、必要なときに取り込む

- AGENTS.md（オープンな指示書の形式） (https://agents.md/)