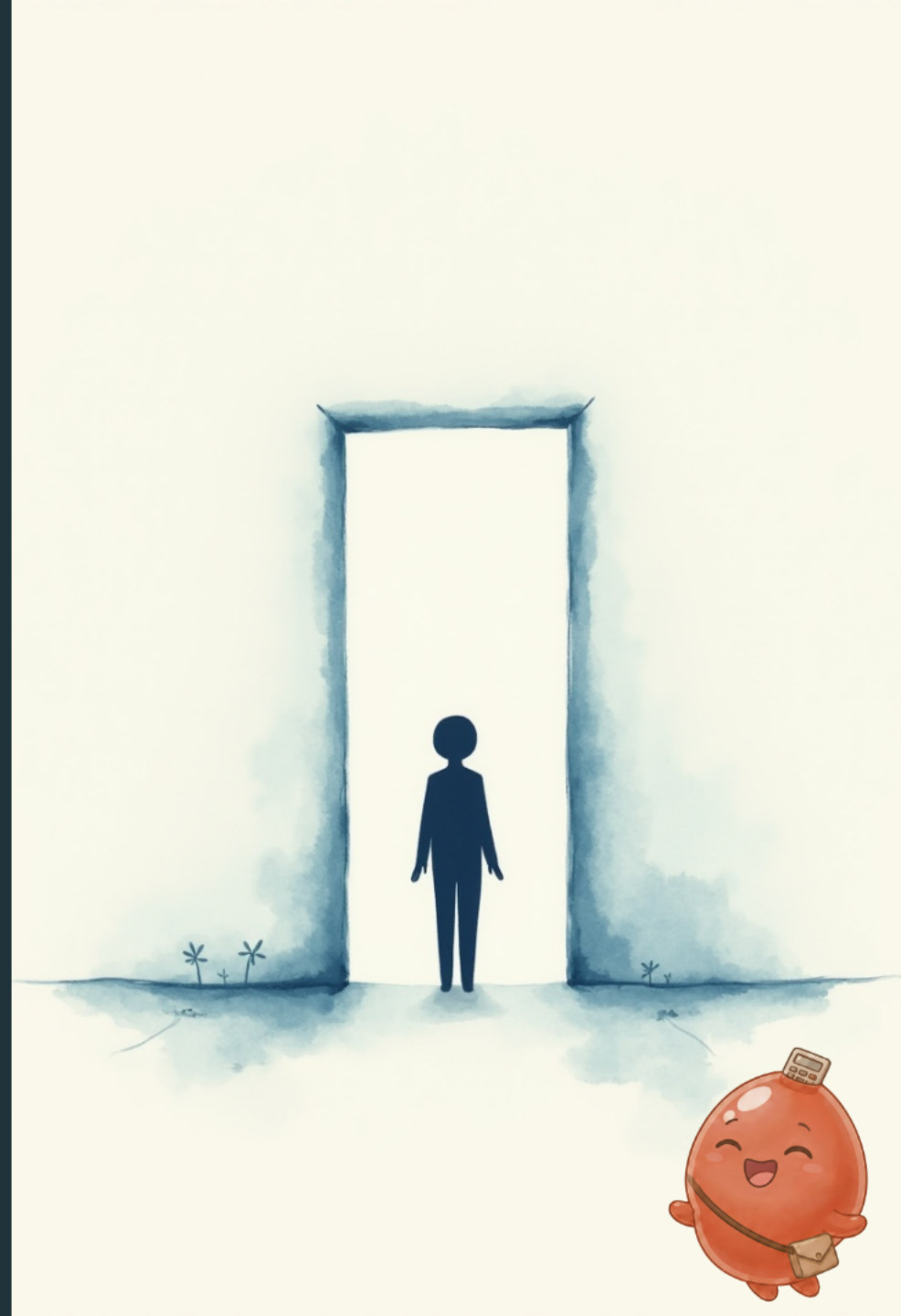


次の扉： 自走

ここまでの旅の、その先を覗く回

AIユーザーレベルチェック（無料）→

ai-sozo.com/tools/ai-user-level



「自走」だけ、まだ点数がついていない

委任・検証・止めどき・言語化は仕上げ、実際の環境にも置けた。自走だけは、まだ「これが正解」というやり方が決まっていない領域。

第1回の総合判定にも、自走は含まれていなかった。

今日は、その先にある景色を少しだけ覗く回。ここから先は、まだ答えの出ていない領域の話。



一本道だと、行き止まりかどうか確かめようがない

実行して、検証して、直す。1つの答えに向かって進む一本道。うまくいかなくなったとき、本当の行き止まりか、道が悪かっただけかを確かめる方法がない＝局所最適にはまる。

たとえば：複数の道を同時に歩いて、一番良いところまで進めたものを採用する。

1本の線（ループ）ではなく、枝分かれした網の目（グラフ）として作業を捉え直す＝グラフエンジニアリングの入り口。



指示を出したあとの数分、ただ待っていないか

結果が返ってくるまでの時間は、ほとんどが待ち時間。指示を出す瞬間と、確認する瞬間だけが自分の仕事。

たとえ：片方に指示を出したら、その足でもう片方に切り替えて別の指示を出す。戻る頃には最初のほうが終わっている。

Claude Code と Cowork のような、2つ以上のAIを同時に走らせる＝並列運用。慣れると体感の作業量がはっきり変わる（ただし並列に動かす分だけトークン消費も増える）。



同じものを、2つのAIに同時に触らせない

同じファイルを片方が書き換え中に、もう片方も書き換えようとするすると上書き事故になる。とくにGitで管理していない場所や、複数の担当に仕事を割り振るときに起きやすい。

気をつけるだけでは防ぎきれない。実は、Claude Codeにはこれに対する専用の機能がある。



公式の逃げ道：ワークツリー

セッションごとに別々の作業場所を用意する仕組み。ここで書き換えたファイルは、別のセッションから絶対に触れない（触ろうとしてもその場で断られる）。

使い方：`claude --worktree <名前>` で起動するだけ。バックグラウンドセッションは、ファイル編集の直前に自動で隔離された場所へ移動する。

「触る場所を分ける」という気配りは、今は補助的なもの。まずはワークツリーに頼る。



「グラフエンジニアリング」は、まだ定まった言葉ではない

この呼び方自体は、業界で使われはじめた言葉。Claude Code公式の決まった名前ではない。

本来はもっと広い話（複数のループの組み合わせ方・結果の選び方・止め方まで設計する考え方）で、今日話したのはその一部。

同じ課題に複数の道筋を同時に走らせ、一番良い結果を選ぶ発想。一本のループでは気づけない答えに、たどり着ける可能性がある。



きっかけが「自分」から「外の出来事」に変わるとき

そのさらに先には「自走」という景色。実行を始めるきっかけが、メールが届いた・時間になった、といった外部の出来事変わったとき、それはこの講座の範囲を超えた領域になる。

この先はまだ、誰にとっても正攻法が定まっていない。焦る必要はない。そういう方向があると知っておくだけで、今は十分。



ループの中で手を動かす人から、 ループの外側にいる設計者へ

委任を広げ、検証を仕組みにし、止めどきを決め、それを言葉にして、実際の環境に置いた。

ここまで来られたなら、それはもう達成されている。



先の話より、今の仕組みを使い込むこと

グラフエンジニアリングや自走は、まだ先の話でいい。今は、ここまで作ってきた仕組みを、実際の仕事の中で使い込んでいくことのほうが、ずっと大事。



お疲れさまでした

自分の現在地を測るところから始まった、全7回。
委任・検証・止めどき・言語化を仕上げ、実際の環境に置いた。

- 「自走」がまだ正攻法の定まっていない領域だと理解した
- グラフエンジニアリングの入り口を知った
- 待ち時間を使った並列運用を、実際に1つ試した
- 「ループの中」から「外」へ移動したと実感した

次は第8回、任せる範囲を安全に広げるための設定の話です。

音声について

本サイトの教材動画のナレーションには、
VOICEVOXの音声合成ソフトウェアを使用しています。

いくらちゃん：四国めたん

ちびた：ずんだもん

