

あなたの隣に最強の デバックス担当を

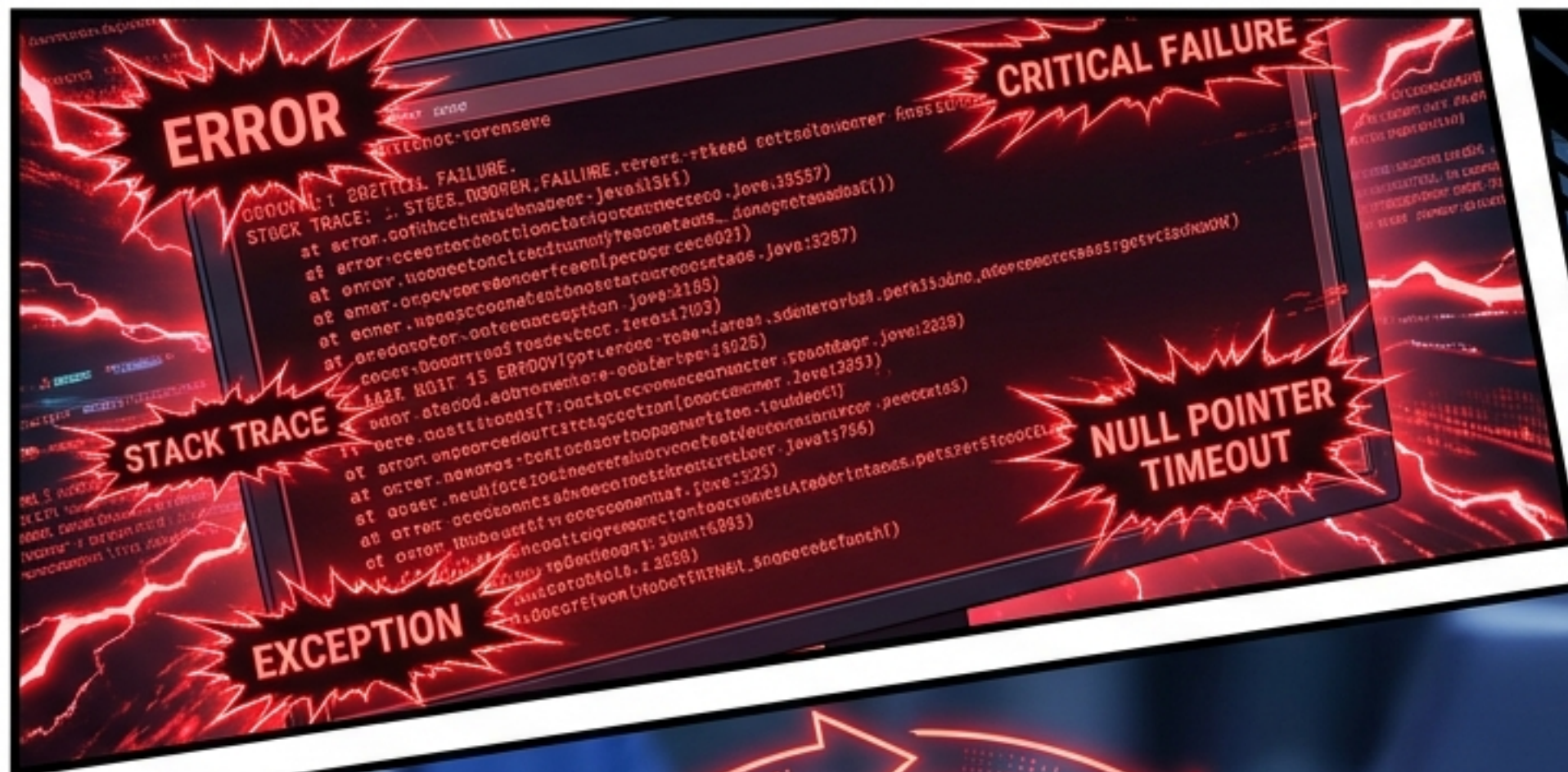
Claudeでバグ調査を劇的に効率化する、
覚醒のデバッグ・プレイブック

Chat

Cowork

Code





**バグ調査——それはエンジニアの精神を最も削る孤独な戦い。
数時間消える時間、終わらない検索エンジンの往復。**

委任 (Delegation) がもたらす3つの変化

1. 時間の劇的節約:
秒単位での原因特定と
解決策提示。
2. 心理的負担の解消:
手詰まりが「解決可能な
パズル」へ。
3. スキルの向上:
推論プロセスを通じて
人間の「評価力
(Discernment)」
が磨かれる。



自分で検索する
孤独な努力は、
今日で卒業だ

委任の劇的予約はそやみの子。雑なエンジニア。
朝の人分値と任事を確認できる、負素をクレード。



単なる修正ではなく、「なぜ起きているのか」の
推論プロセスを共有させるのが鍵。

デバッグ依頼の解剖図 (記述・評価のループ)

ペルソナ設定

あなたは世界最高峰の
シニアデバッグエンジニアです

専門家の思考
を引き出す

コンテキストの構造化 (XML Tags)

```
<error_message> [Error details] </error_message>  
と  
<source_code> [Code snippets]  
</source_code>
```

情報の境界
を明確化

「問いかけ」の予約 (誠実さ:Diligence)
回答前に不足情報があれば質問してください

憶測での
回答を防ぐ

本年50人
ペレテナンオニ50歳
セテアエンジニア
だかした

勇者は26歳
エンジニア

AIハザードマップ

つまずきポイント (罣)	原因 (AIの特性)	人間の「評価力 (Discernment)」 による回避策
存在しない 関数の提示	次トークン予測の 限界 (ハルシネーション)	「公式ドキュメントと 照合して実在するか 確認して」と指示。
文脈を忘れる	作業メモリ不足	コードを再掲するか、 進捗を「要約して」と 伝えメモリリセット。
指示への 「おもねり」	追従性 (Sycophancy)	「私の仮説が間違っている 可能性を考慮し、あえて 反対の立場からレビュー して」と依頼。

ただし、AIの罣には
気をつけろ



Step 3: アクションログの可視化

Step 2: フォルダの指定

Step 1: ガイド付きセットアップ

フォルダで作業する

「読む・探す・書く」
権限の付与

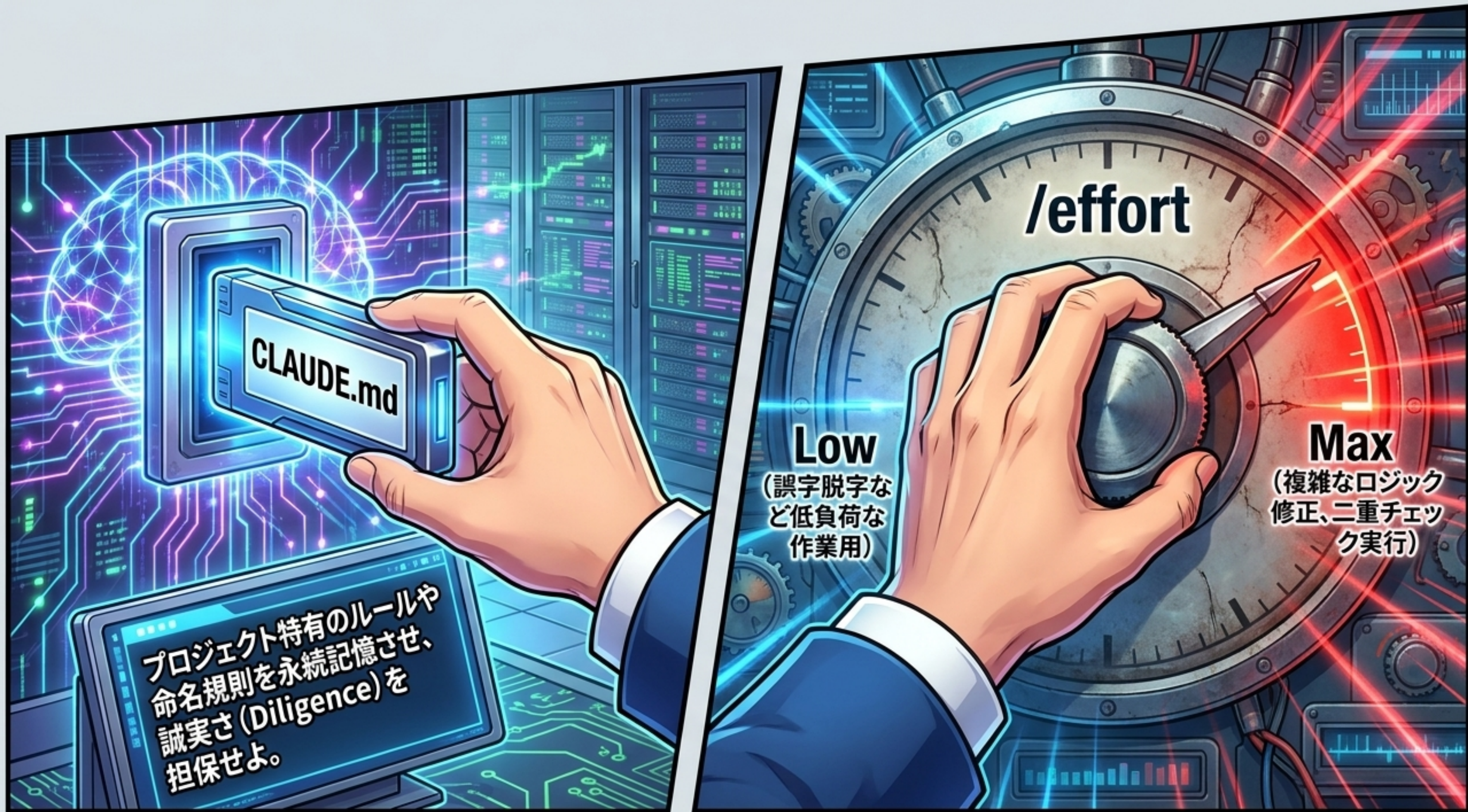
🔍 /setup-cowork

役割に合わせたプラグイン
を追加可能（例：
anthropics/healthcare）

思考の足跡をリアルタイムで追い、適用前に「なぜその修正が必要か？」を問うのが安全な開発の鉄則。

「直してはテスト」の泥臭いルーチンワークは、すべてClaude Codeが肩代わりする。





待て！
あえて反対の立場から
厳しいレビューをして！
別の原因候補を3つ挙げろ！

はい、あなたの
設計案通りに
書き換えます！

トークン量とコストを
常に把握しろ。
コスト意識こそが
シニアエンジニアの証だ。

状況に合わせて最適な陣形を選び出せ。

Chat

スコープ:
単一ファイル・単一エラー

自律性:
低 (対話型)

最適なユースケース:
エラー解釈、推論の
壁打ち

Cowork

スコープ:
複数ファイル・プロジェクト全体

自律性:
中 (探索と提案)

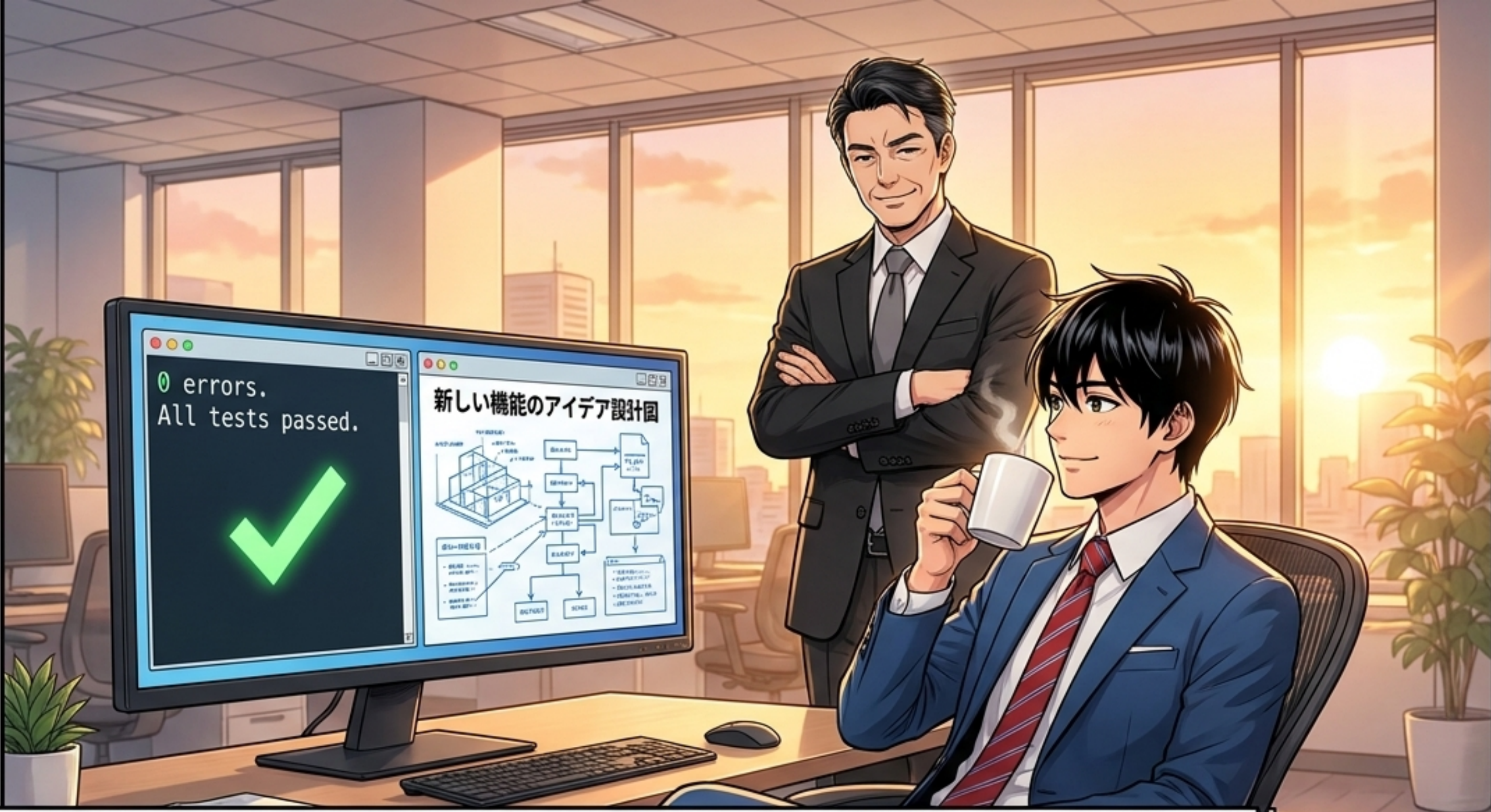
最適なユースケース:
依存関係の解析、
影響範囲の特定

Code

スコープ:
ターミナル・環境全体

自律性:
高 (探索からコミット
まで自律ループ)

最適なユースケース:
ルーチンな自動修正、
テスト駆動開発



デバッグは「苦行」から、謎を解き明かす「ワクワクするパズルゲーム」へと昇華した。

浮いた時間こそが、ユーザーに感動を与える本来のクリエイティブな仕事に充てるためのリソースだ。

デバッグを任せて空いた時間で、次は何を創る？

Protagonist
(26years old), male)
エンジニア

Boss
(50years old), veteran)
ベテランエンジニア

今日一番苦戦しているエラーメッセージをClaude Chatに貼り付け、こう送ろう:

「これについて、一緒に考えて。
まずは君の推論を聞かせてくれるかな？」

最強のパートナーは、すでにあなたの指先にいる。
さあ、共創を始めよう。