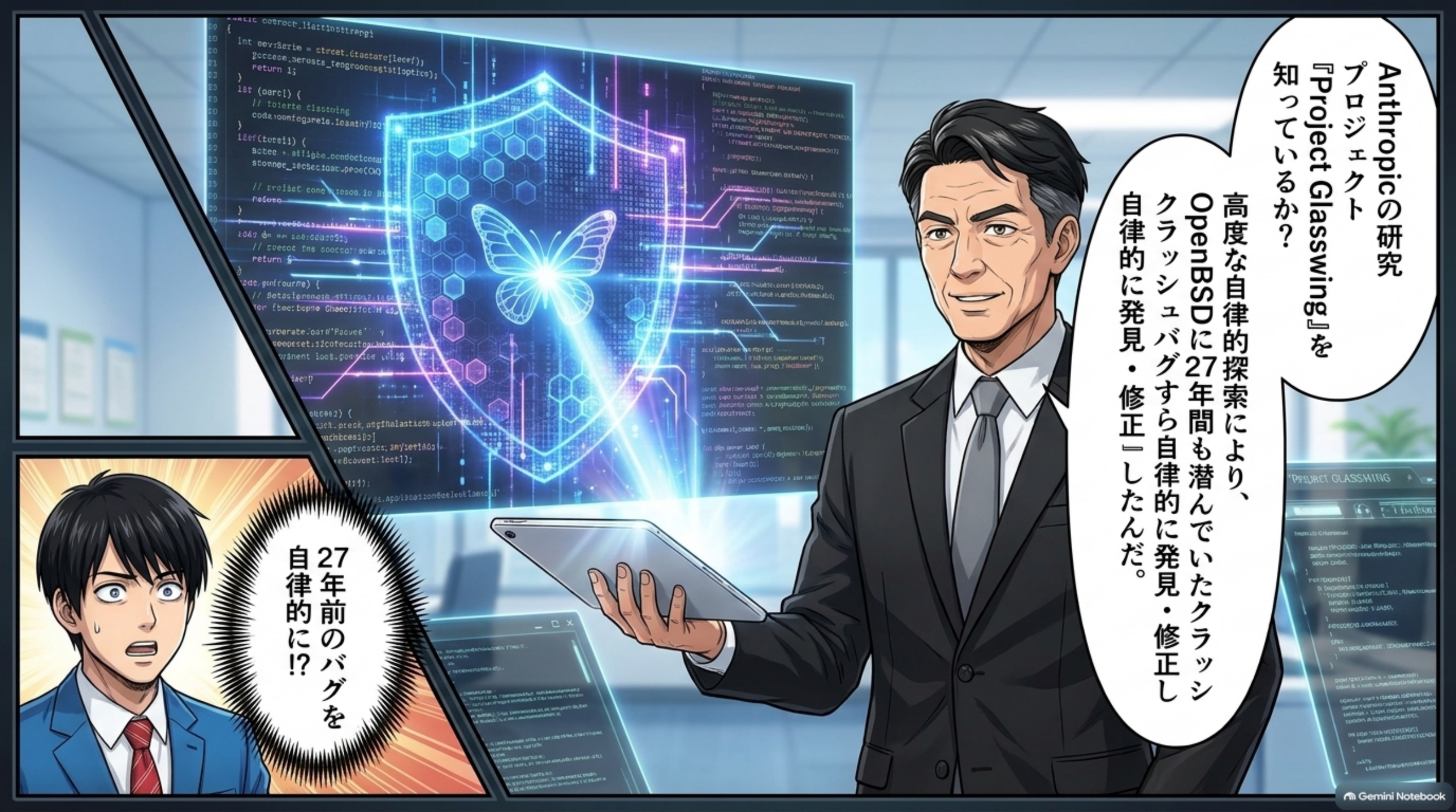
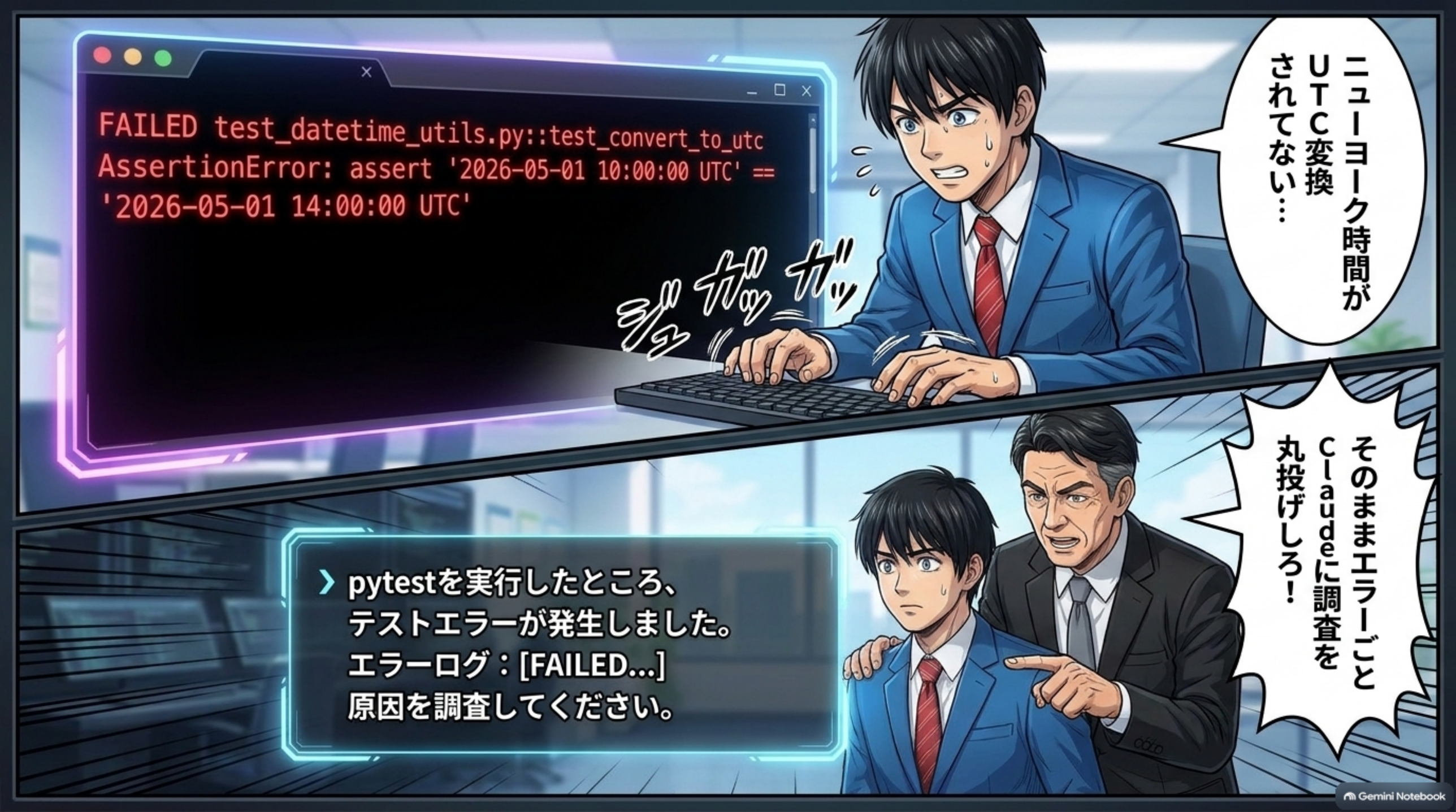

もう何時間もログと格闘するのはやめにしよう
AIにバグ調査を丸投げ！
Claude Codeで変わる新・開発体験
Gemini Notebook


原因不明のエラー。手動の検索。
デバッグは開発で最も
精神力を削る作業だ。
なぜテストが
通らないんだ…
もう3時間ログと
格闘してるぞ…
おいおい、
まだ人間が1行ずつ
ログを読んでるのか？

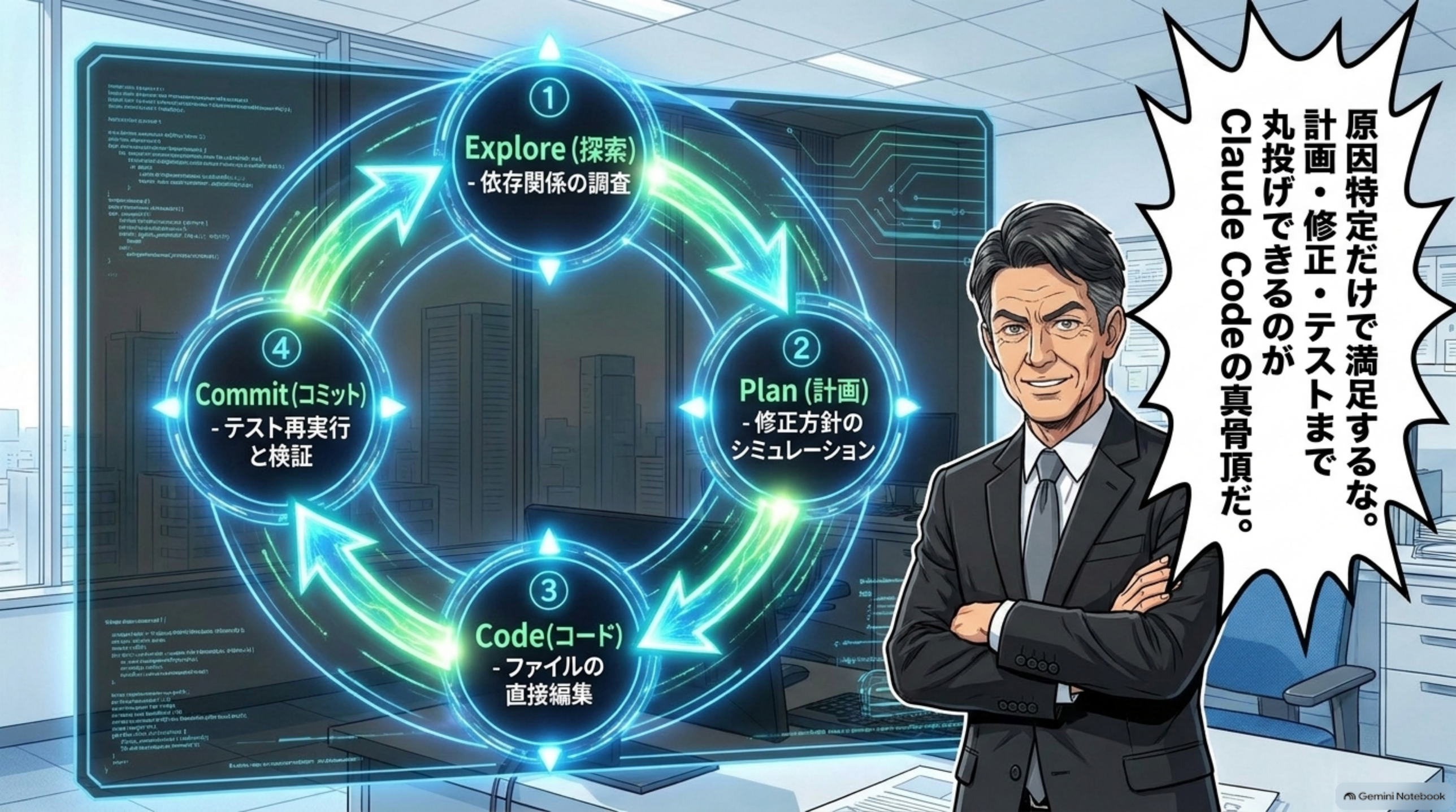

Anthropicの研究プロジェクト『Project Glasswing』を知っているか？
高度な自律的探索により、OpenBSDに27年間も潜んでいたクラッシュバグすら自律的に発見・修正し自律的に発見・修正』したんだ。
27年前のバグを自律的に!?
PROJECT GLASSWING


原因特定が数分!?
しかも検証まで自動!?
従来のデバッグ
Claudeデバッグ
数時間～数日
わずか数分
人間が1行ずつ解読
コード全体から自律特定
手動テスト
自動検証
疲労大
スムーズな対話
準備はこれだけだ
Claude Account
Terminal
Python/pytest
Code Editor


FAILED test_datetime_utils.py::test_convert_to_utc
AssertionError: assert '2026-05-01 10:00:00 UTC' ==
'2026-05-01 14:00:00 UTC'
カタカタ
ニューヨーク時間がUTC変換されてない…
そのままエラーごとClaudeに調査を丸投げしろ！
› pytestを実行したところ、
テストエラーが発生しました。
エラーログ：[FAILED...]
原因を調査してください。

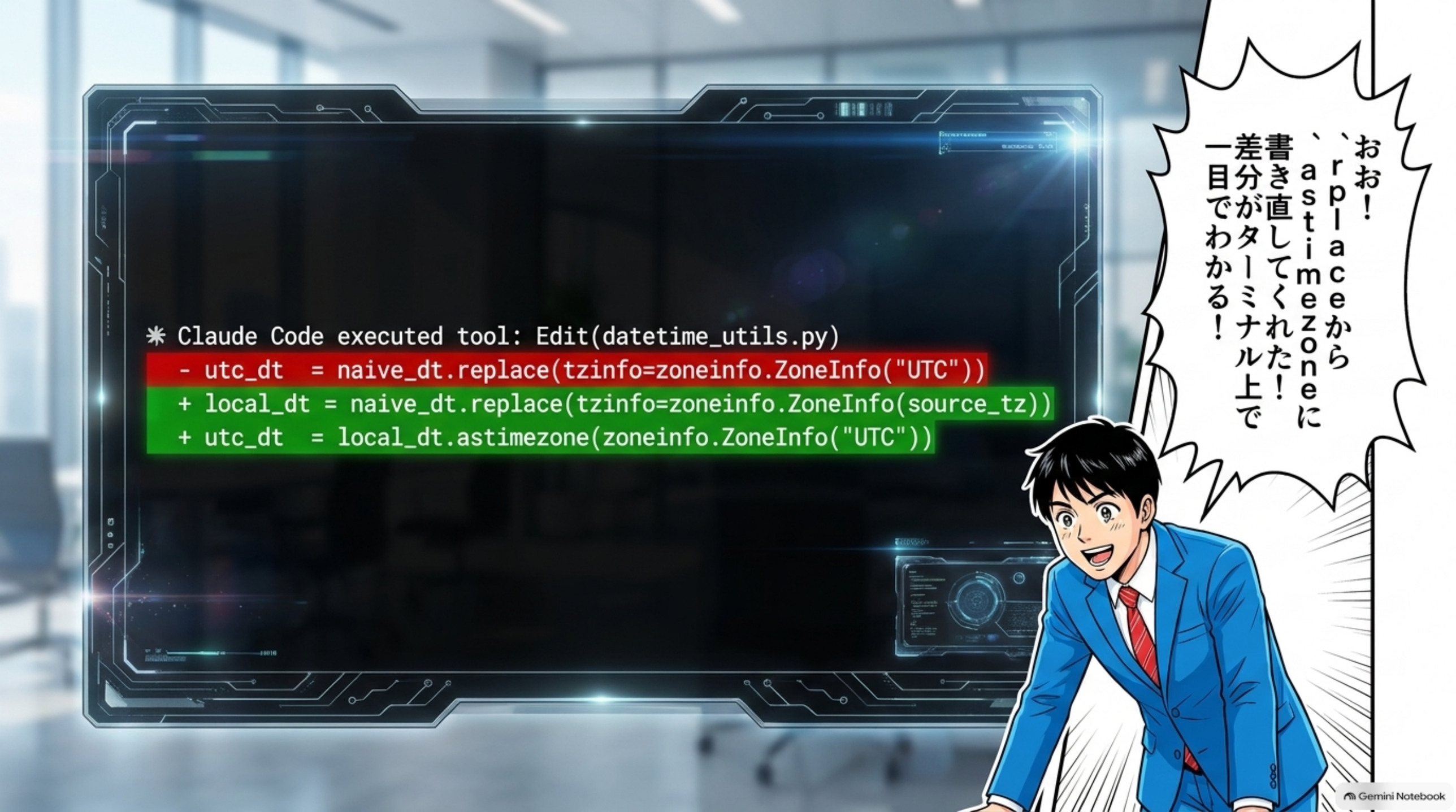

* Claude Code is thinking...
Read(datetime_utils.py)
Read(test_datetime_utils.py)
原因：タイムゾーンのオフセット変換が行われておらず、単にラベルを付け替えているだけです。
`naive_dt.replace(tzinfo=ZoneInfo("UTC"))`が問題です。
勝手にファイルを読み込み始めた!?
しかもこれだけで根本原因がわかった！
Gemini Notebook


あれ、なんかAIの動作が重くなってきた…？過去の指示も忘れてる気が…
『Lost in the Middle(中間での迷子)』状態だな。作業メモリの管理が必要だ。
/context：メモリ使用量の確認
/compact：履歴を圧縮して枠を空ける
/clear：完全にリセットする
Gemini Notebook

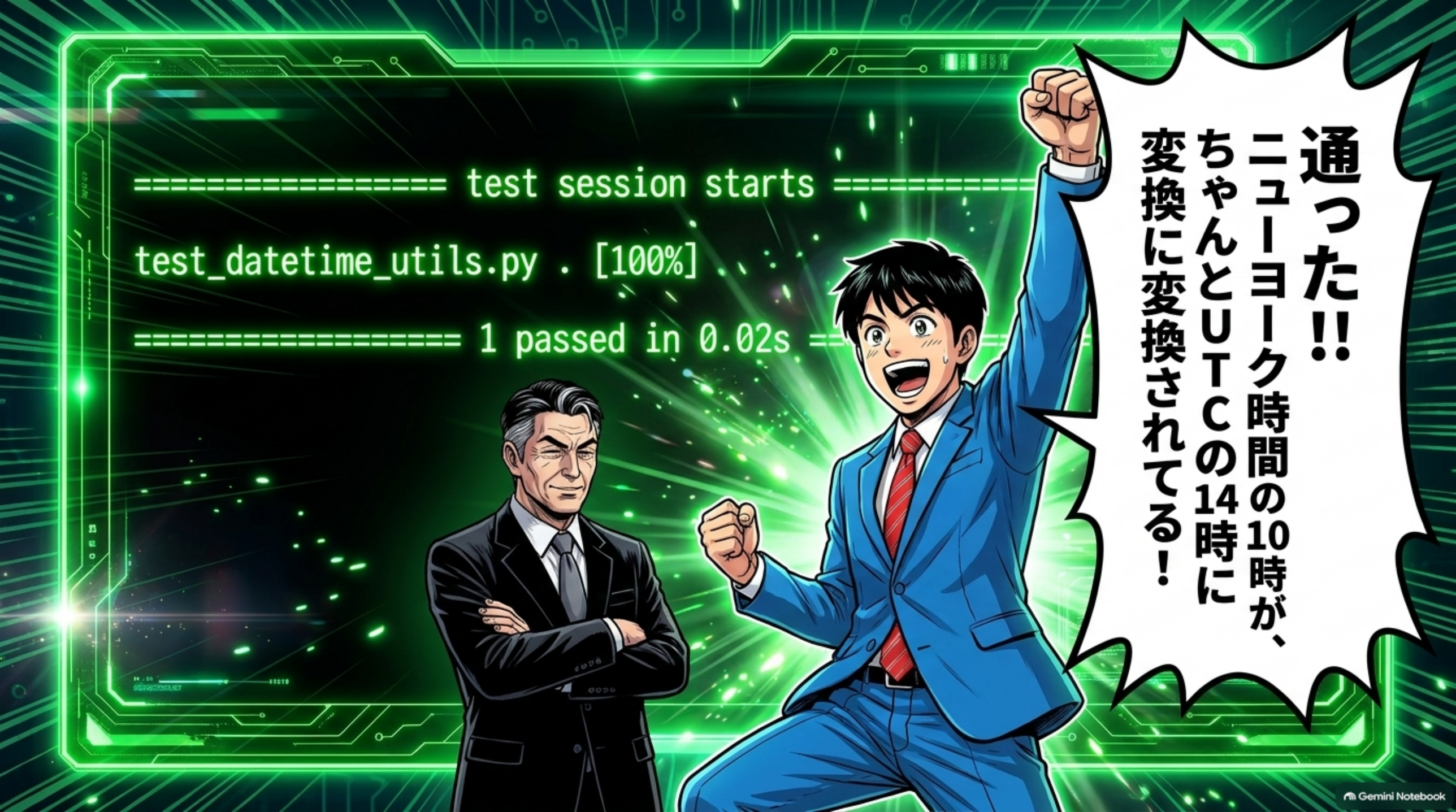

①
Explore (探索)
- 依存関係の調査
②
Plan (計画)
- 修正方針のシミュレーション
③
Code(コード)
- ファイルの直接編集
④
Commit(コミット)
- テスト再実行と検証
原因特定だけで満足するな。
計画・修正・テストまで
丸投げできるのが
Claude Codeの真骨頂だ。
Gemini Notebook

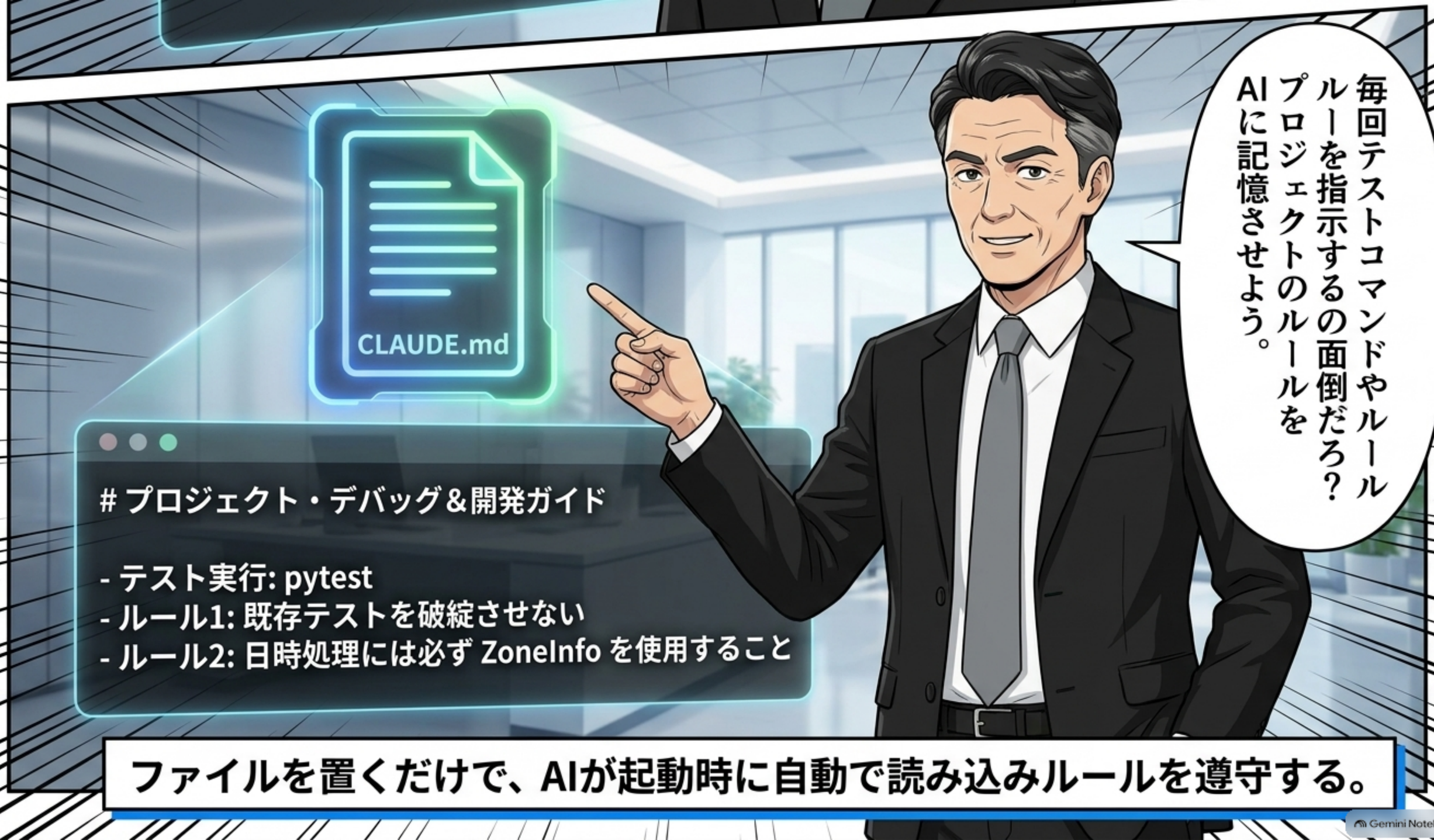

【Anthropic The AI Fluency Index】
AIの洗練された出力を見ると…
・推論への疑問提示：-3.1%pt
・ファクトチェック：-3.7%pt
ただし、注意しろ。
AIの出力が綺麗だからと
鵜呑みにする
『アーティファクト効果』に
に陥るなよ。
必ず人間が Diff を確認し、
反復（Iteration）しろ。
対話によって推論への疑問提示は
5.6倍 に跳ね上がる！

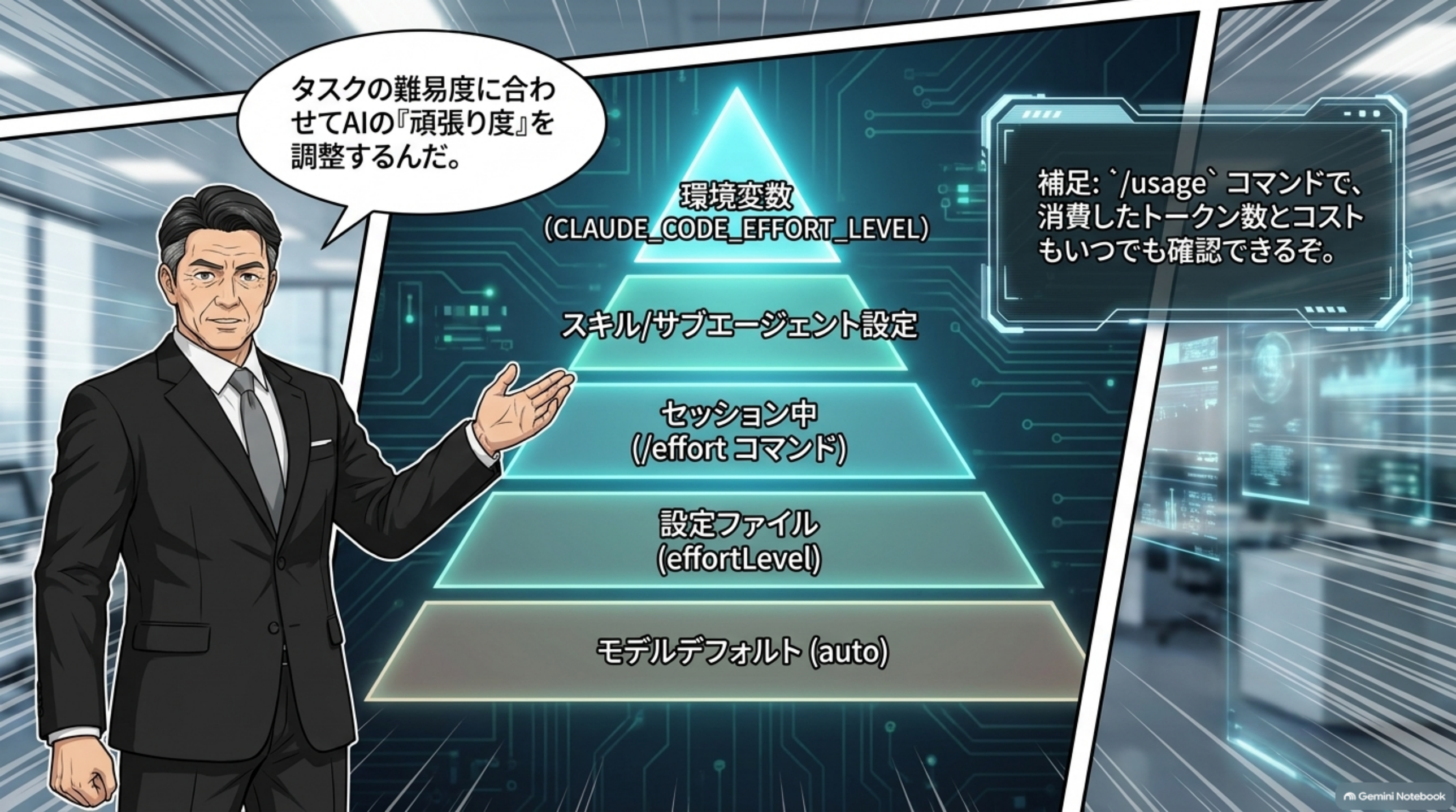
```
✳ Claude Code executed tool: Edit(datetime_utils.py)
  - utc_dt   = naive_dt.replace(tzinfo=zoneinfo.ZoneInfo("UTC"))
  + local_dt = naive_dt.replace(tzinfo=zoneinfo.ZoneInfo(source_tz))
  + utc_dt   = local_dt.astimezone(zoneinfo.ZoneInfo("UTC"))
```



Gemini Notebook


========== test session starts ==========
test_datetime_utils.py . [100%]
========== 1 passed in 0.02s ==========
通った!!
ニューヨーク時間の10時が、
ちゃんとUTCの14時に
変換に変換されてる！


毎回テストコマンドやルールを指示するの面倒だろ？
プロジェクトのルールをAIに記憶させよう。
CLAUDE.md
プロジェクト・デバッグ＆開発ガイド
- テスト実行: pytest
- ルール1: 既存テストを破綻させない
- ルール2: 日時処理には必ず ZoneInfo を使用すること
ファイルを置くだけで、AIが起動時に自動で読み込みルールを遵守する。


タスクの難易度に合わせてAIの『頑張り度』を調整するんだ。
環境変数
(CLAUDE_CODE_EFFORT_LEVEL)
スキル/サブエージェント設定
セッション中
(/effort コマンド)
設定ファイル
(effortLevel)
モデルデフォルト (auto)
補足: `/usage` コマンドで、消費したトークン数とコストもいつでも確認できるぞ。
Gemini Notebook

症状:
途中で作業を
打ち切る / 未完成

対策:
`/effort` を上げる
(low -> medium/high)

症状:
簡単な作業なのに
何分も思考する

対策:
`/effort` を下げる
(過剰思考を防ぐ)

症状:
存在しない関数を
推測して使い続ける

対策:
`/model` で高機能モデルへ変更
(知識不足)



⚠ 鉄則: モデル変更時はエフォートを一度「auto」に戻すこと！

Gemini Notebook


さあ、
次は君の番だ。
これで本来
これ作りたかった
新機能の開発に
集中できます！
今すぐターミナルを開き、エラーログを貼り付けてみよう。
› このエラーの原因を調べて、修正案を教えてください：[ここにエラーログを貼り付け]
Gemini Notebook