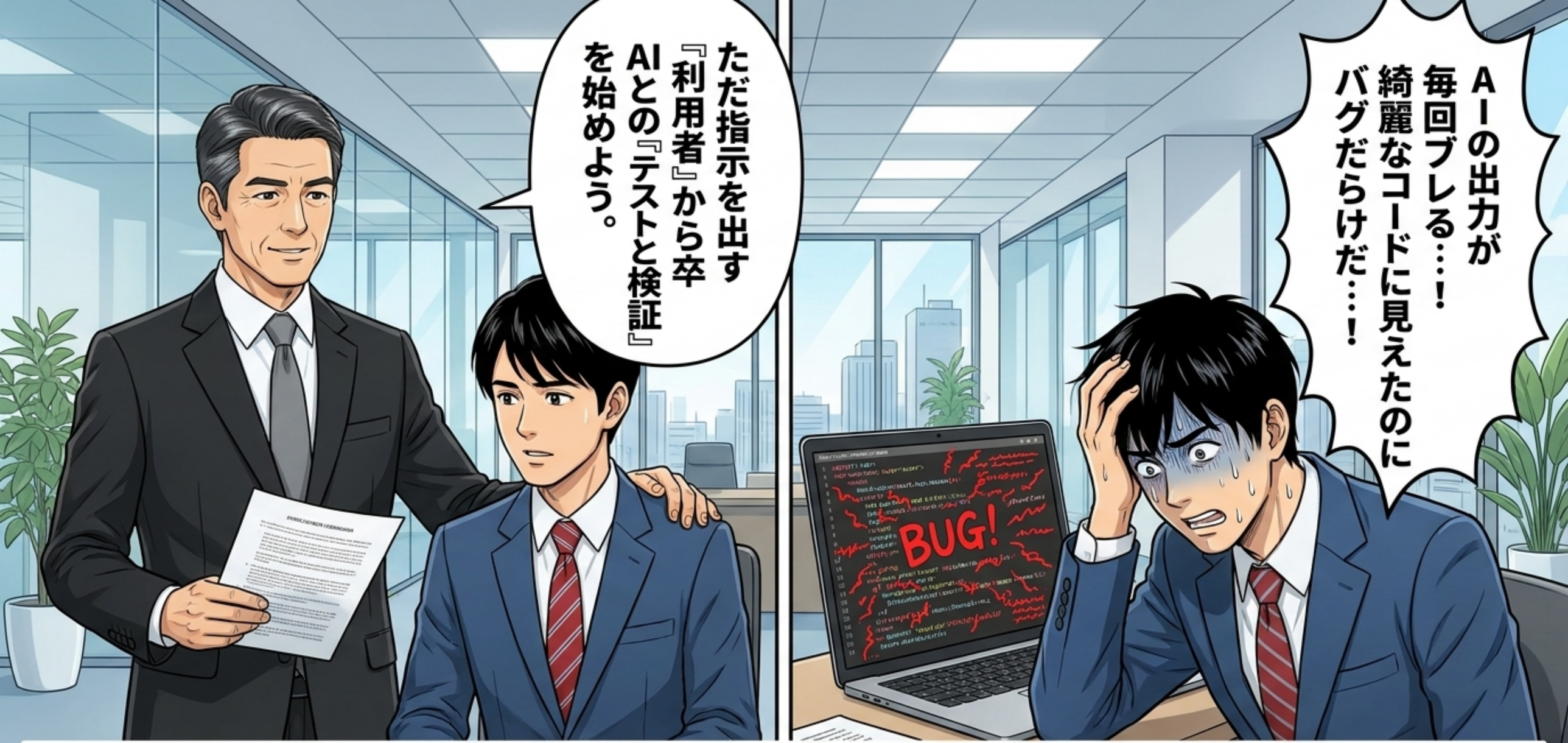


Claudeで始める「テストの作成と実行」完全ガイド

Gemini Notebook


4D AI Fluency（AI活用力）
フレームワークにおける
「テスト」の意義
The Significance of "Testing" in the 4D AI Fluency Framework.
倫理的責任
(Diligence)
評価力
(Discernment)
記述力
(Description)
委任力
(Delegation)
Anthropicが提唱する4つの力だ。
君は今『委任』と『記述』にとどまっている。
品質を担保する『評価力』と『倫理的責任』
へ登る鍵が、テストの設計だ。
These are the four powers proposed by Anthropic.
You are currently at 'Delegation' and 'Description'.
The key to climbing to 'Discernment' and 'Diligence'
to ensure quality is test design.
Gemini Notebook


ステップ3：
Claude Codeによる自律ループ
ステップ2：
テストの実行と自動評価
ステップ1：
テストデータセットの自動生成
Gemini Notebook


本番環境は甘くない。
Claudeの言語生成力を使い、
人間が思いつかない多様な
『エッジケース』を
数分で大量生成させるんだ。
Glaring typos,
you'se your ft...
Hüo achet!!
glaring typos..!
人間が作ると
「思いつきやすい正常系」に偏る。


なるほど、XMLタグとJSON Schemaで型を定型を指定すれば、プログラムで処理しやすい検証データが一瞬で一瞬で完成するんですね…！
「JSON Schema形式に従って出力してください。解説や前置き文章は一切含めず、純粋なJSONデータのみを出力してください。」
/ <test_cases>
{
"test_cases": [
{
"name": "正常系",
"type": "TC-正常系",
"description": "概要",
"datatype": "TC-001 (正常系)"
},
TC-001 (正常系)
},
TC-003 (感情的な抗議 - 指定日に商品が届かないんだけど！)
{
"name": "TC-001",
"type": "TC-000",
"description": "概要",
"datatype": "指定日に商品が届かないんだけど！)"
}
]
}
Gemini Notebook


罠1：「次トークン予測 (Next-Token Prediction)」―
モデルは無難で確率の高い文章に偏る。
DANGER!
WARNING!
Test Case 1
Test Case 2
Test Case 2
Test Case 3
【解決策】
ペルソナ、感情値、
誤字の有無Oの有無など
「評価軸」を
プロンプトで明示せよ！
BLAM!
罠2：ハルシネーションと形式エラー。
GLITCH!
{
"test_cases": [
{ "name": "正常系",
"type": "正常系",
"description": "網羅",
"datatype": "TC-001 (正常系)"
},
{
"name": "TC-001",
"type": "TC-000",
"description": "網羅",
"datatype": "指定日に周期が曜がないんだけど！"
}
]
}
ERROR!
【解決策】
「Few-Shotプロンプティング」―
望ましい正解例を1～2件
プロンプト内に提示せよ！
FIX IT!
Gemini Notebook


気をつけろ！
『アーティファクト効果』だ。
見た目が綺麗に整っていると、
人間は批判的思考を停止して
しまう！
バッ
見た目の整った出力に対し、人間の事実確認・推論のチェック能力は
統計的に低下する（-3.1〜-5.2ポイント）。だからこそ「自動評価」が必須だ。
Gemini Notebook

コードベース採点
(Code-Based Grading)



プログラムが「文字数・必須須キーワード・JSON構造」をTrue/Falseで絶対判定する。

モデルベース採点
(Model-Based Grading)



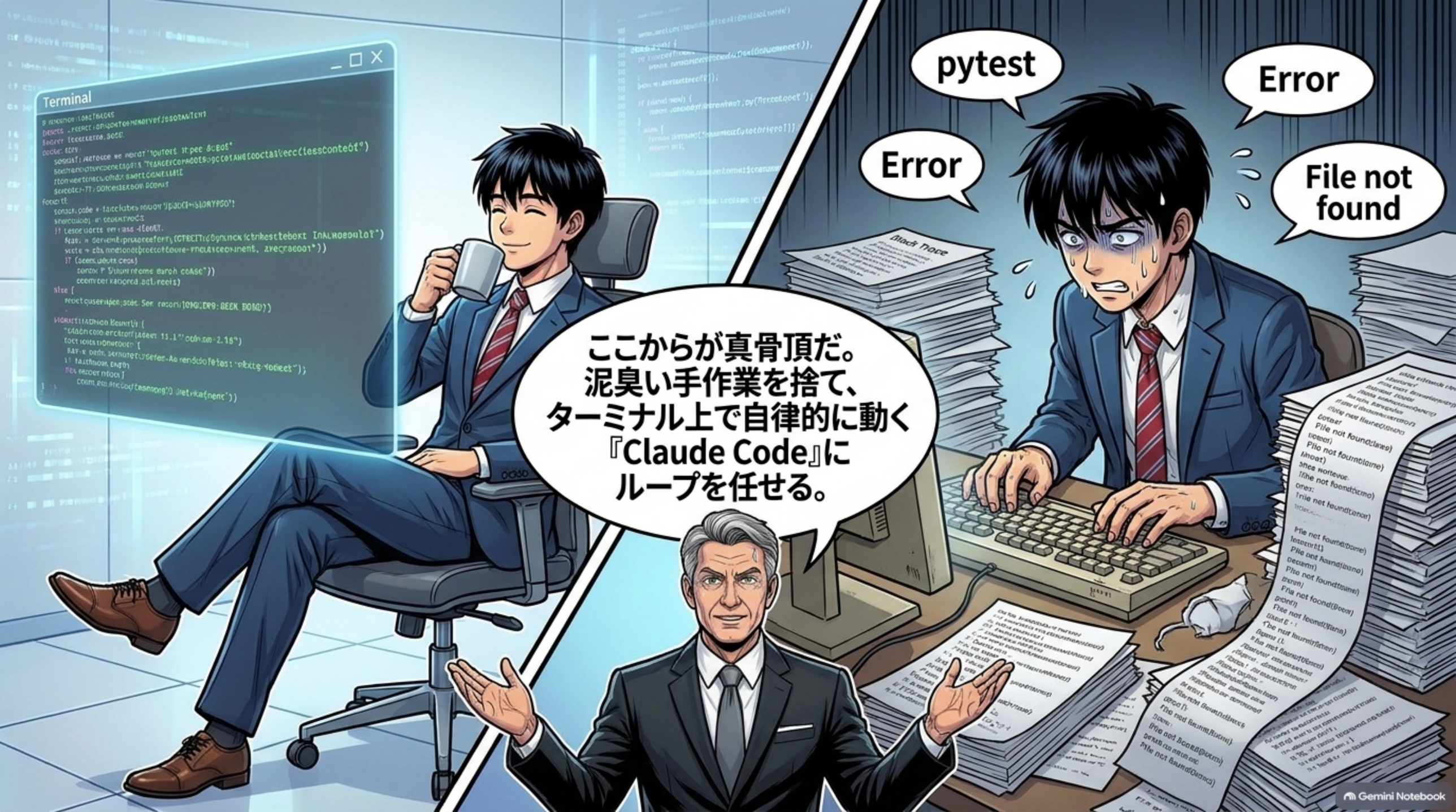
Claudeが「評価ルーブリック」を使い、ニュアンスや論理を1〜5点で定性評価する。

実務では**ハイブリッド構成**が最強だ。
まずプログラムのコードで形式エラーを瞬時に弾き、
パスしたものをClaudeのモデルで深掘り採点する。

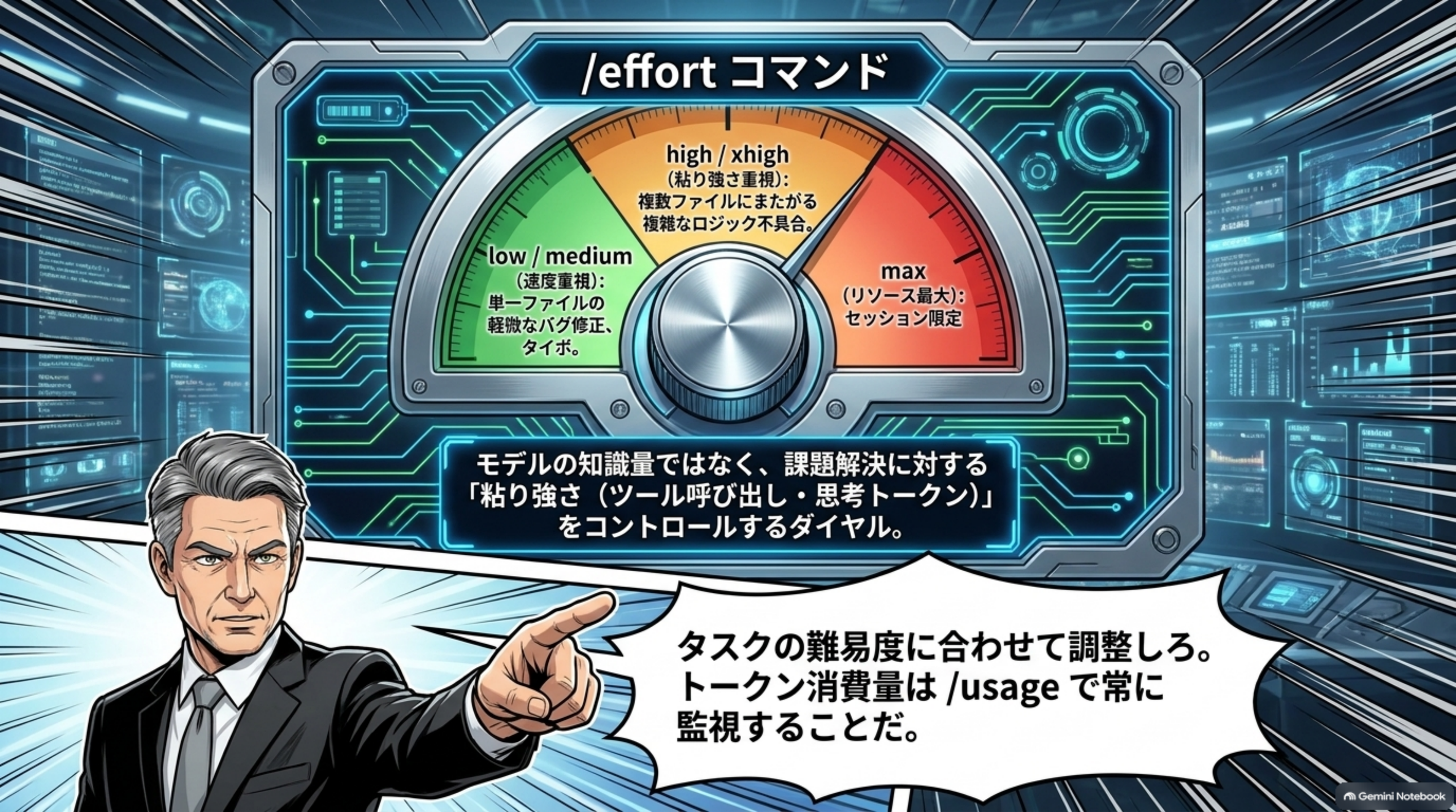
Gemini Notebook


XMLタグで正解と
評価対象を分離しろ。そして『あえて懐疑的で厳格な監査官として振る舞え』とペルソナを上書きするんだ。
Temperature = 0.0
カタカタッカタ！


課題：モデルの「追従性（Sycophancy）」——
人間に忖度して甘い採点をしてしまう。
パパチッ
5/5
Gemini Notebook


Terminal
pytest
Error
Error
File not found
ここからが真骨頂だ。
泥臭い手作業を捨て、
ターミナル上で自律的に動く
『Claude Code』に
ループを任せる。
Gemini Notebook


1
探索 (Explore):
ログとコードの
読み込み
2
計画 (Plan):
原因分析と
修正方針
3
コード (Code):
ソースコードの
自動書き換え
4
コミット (Commit):
変更の確定
5
再テスト (Re-test):
自動検証
失敗
(Fail)
成功
(Success)
エラー原因の分析から、
ファイルの特定、
修正、再テストまで、
Claudeが自律的に
ループを回すんですね
…！
Gemini Notebook


/effort コマンド
low / medium
(速度重視):
単一ファイルの
軽微なバグ修正、
タイポ。
high / xhigh
(粘り強さ重視):
複数ファイルにまたがる
複雑なロジック不具合。
max
(リソース最大):
セッション限定
モデルの知識量ではなく、課題解決に対する
「粘り強さ（ツール呼び出し・思考トークン）」
をコントロールするダイヤル。
タスクの難易度に合わせて調整しろ。
トークン消費量は /usage で常に
監視することだ。
Gemini Notebook

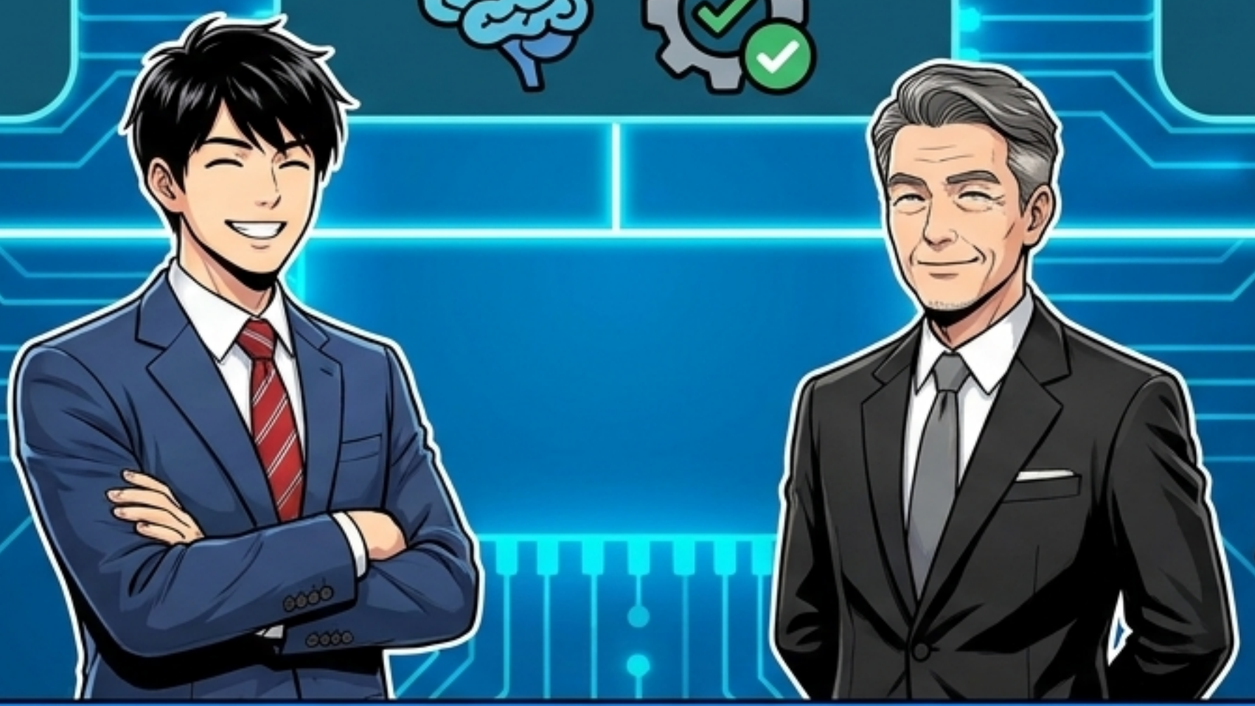

/clear
/compact
COMPRESSED CONTEXT
CLAUDE.md
プロジェクトの
テスト規約や実行手順は
ルートディレクトリの
『CLAUDE.md』に記せ。
セッションをクリアしても
AIの永続的な記憶となる。
Gemini Notebook

マスターパイピロン

【Step 1】
多様なエッジケースを自動生成
(JSON Schema)

【Step 2】
ハイブリッド自動評価
(Model + Code)

【Step 3】
エラー検知時 → Claude Code自律修復ループ



主人 26歳

50歳 父郎

プロンプト改修やシステム変更時のリグレッション（品質劣化）を即座に検知・修復する最強のサイクル。

Gemini Notebook

今すぐ試せる5分間チャレンジ「AIデューデリジェンス声明」

「以下に貼り付ける文章を精査し、ファクトチェックの不備や論理の飛躍を3点、容赦なく批判的に指摘してください。」



直近でAIに作らせた成果物を、あえて批判的に検証させてみよう。
それが「評価力 (Discernment)」を鍛える第一歩だ。

Gemini Notebook