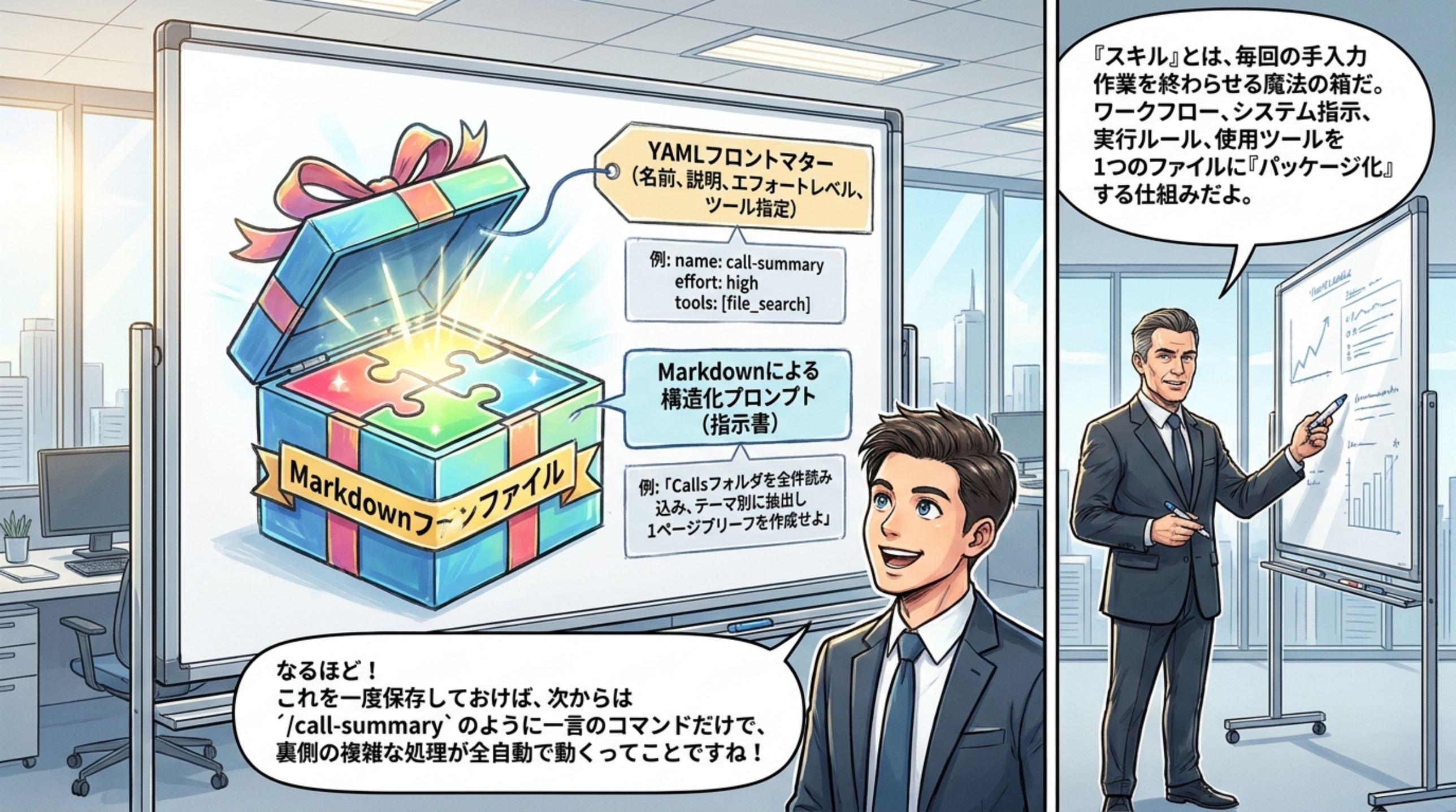
Claude『Skill（スキル）』完全ガイド

～定型業務から解放されるDXバディ・ストーリー～

初心者からチーム活用まで、専門的なAI自動化を完全マスター。



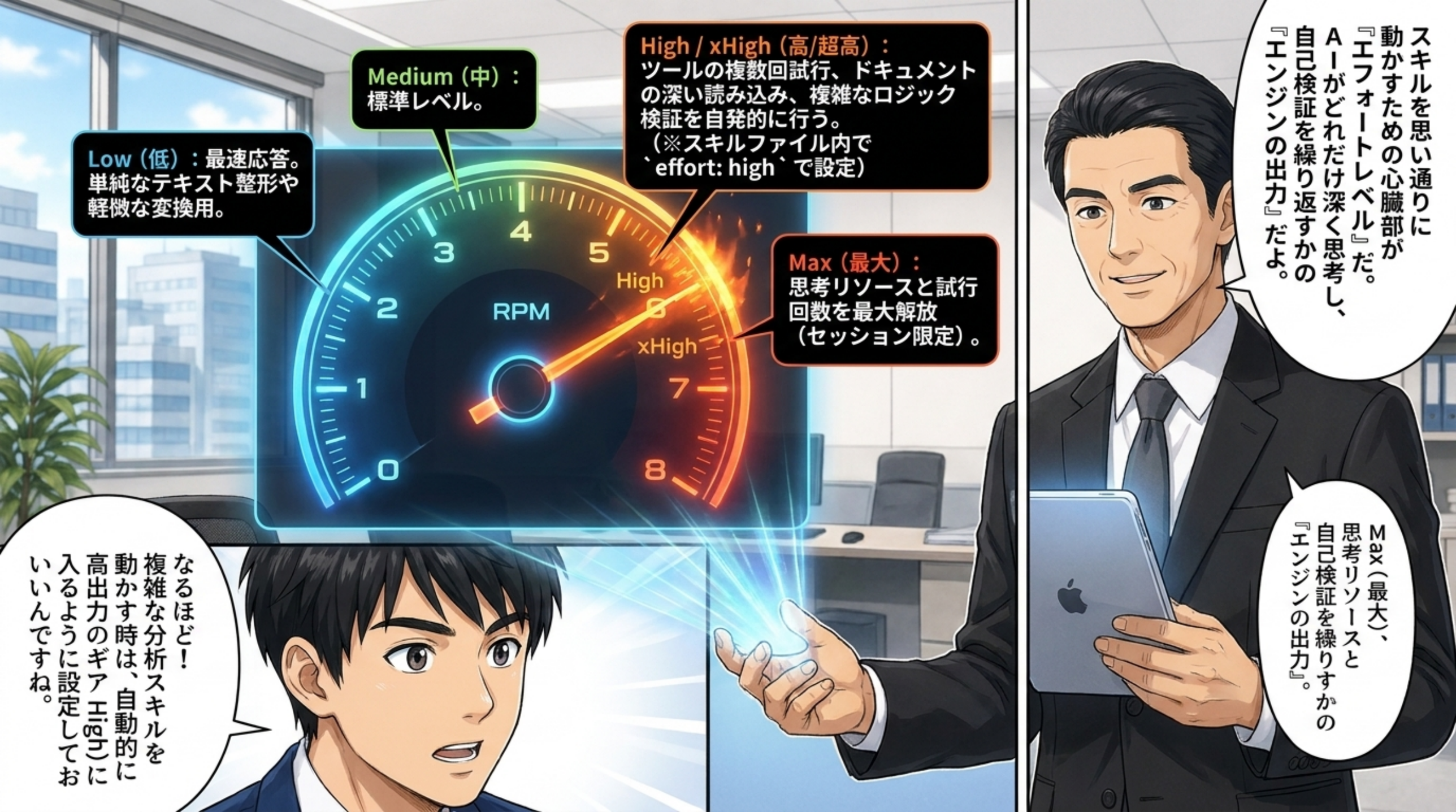


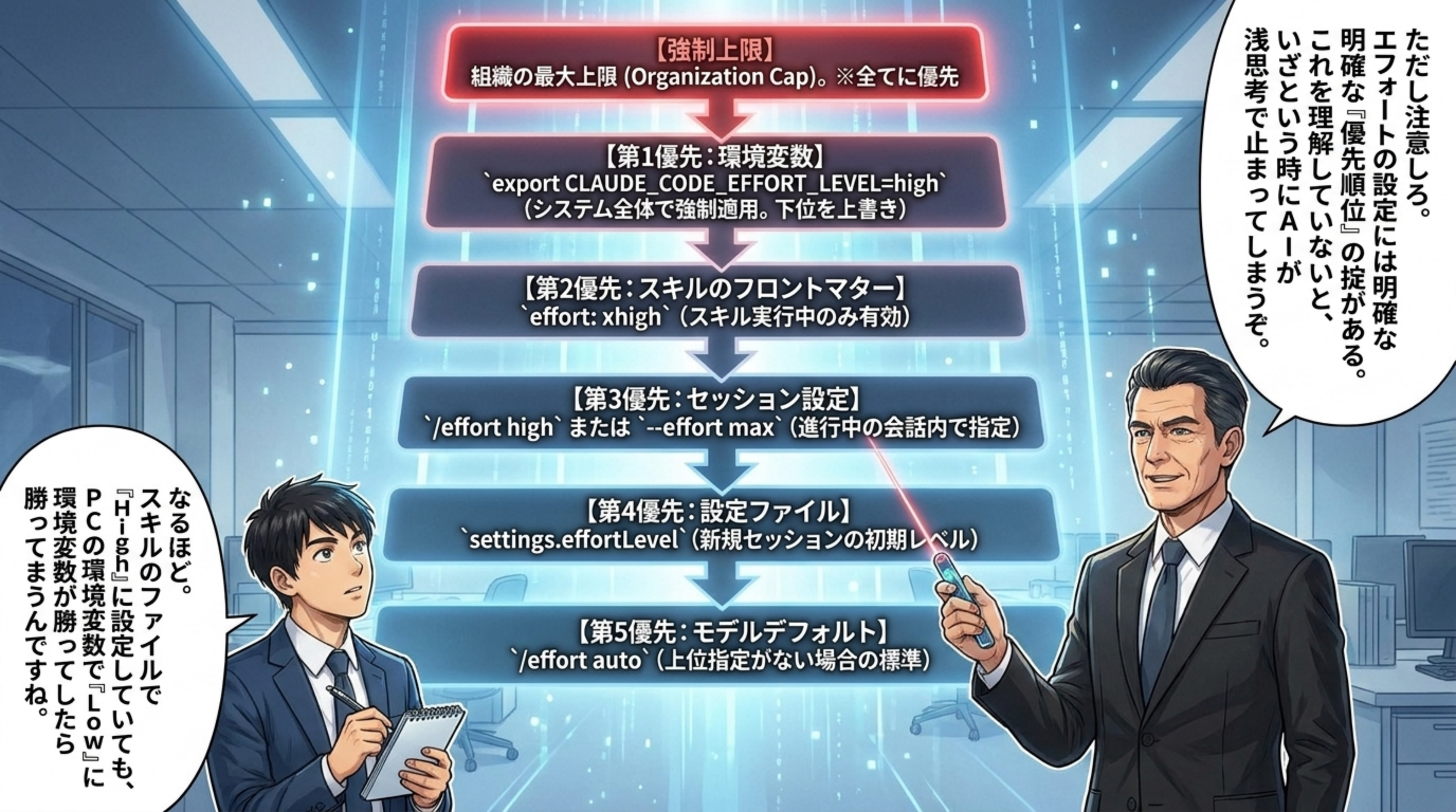

『スキル』とは、毎回の手入力作業を終わらせる魔法の箱だ。ワークフロー、システム指示、実行ルール、使用ツールを1つのファイルに『パッケージ化』する仕組みだよ。
Markdownファイル
YAMLフロントマター（名前、説明、エフォートレベル、ツール指定）
例: name: call-summary
effort: high
tools: [file_search]
Markdownによる構造化プロンプト（指示書）
例:「Callsフォルダを全件読み込み、テーマ別に抽出し1ページブリーフを作成せよ」
なるほど！
これを一度保存しておけば、次からは
`/call-summary` のように一言のコマンドだけで、
裏側の複雑な処理が全自動で動くってことですね！

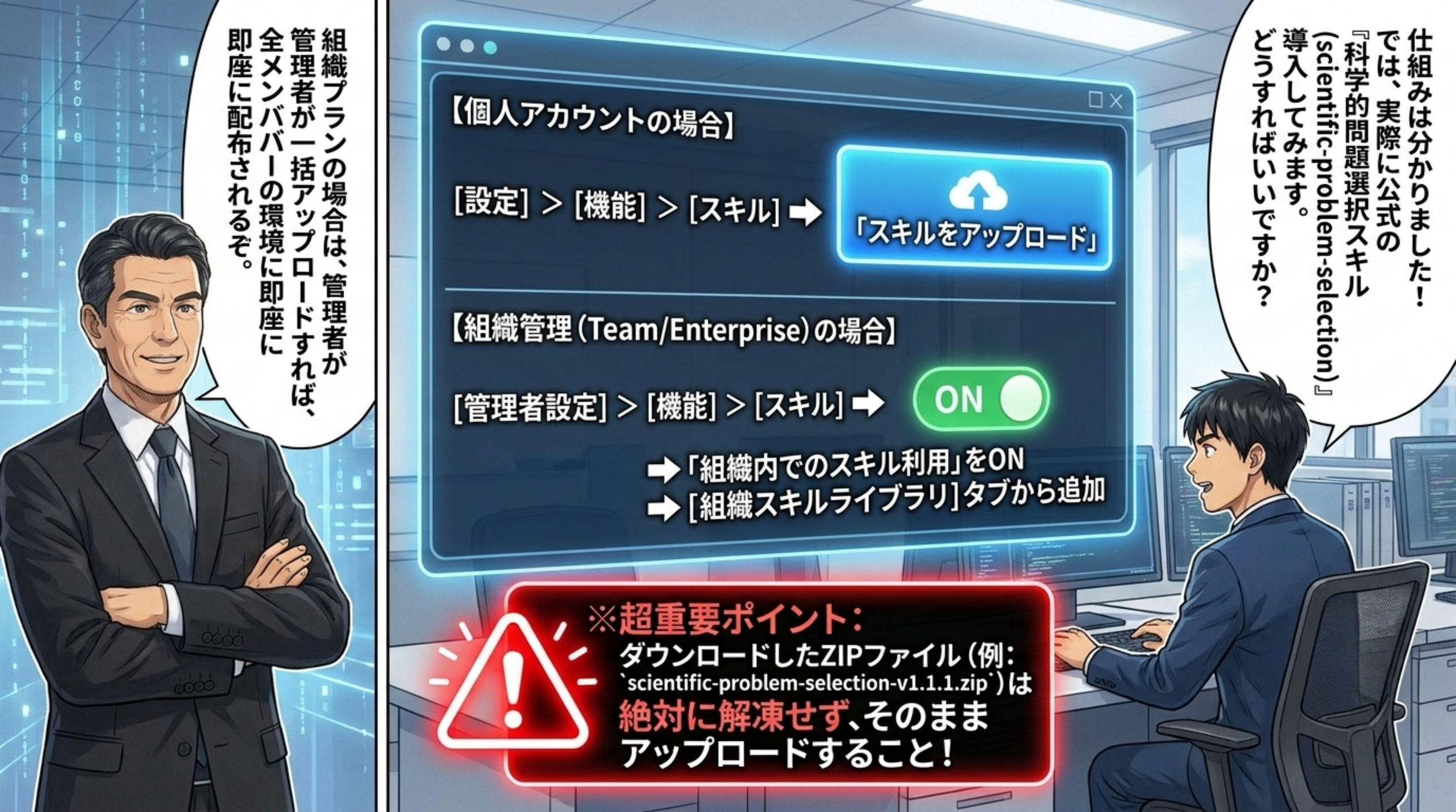




	手動指示の泥臭い運用（Before）	スキルのパッケージ運用（After）
【指示出しの手間】	毎回数百文字の 長文プロンプトを手入力	`/skill-name` や短文コマンドを指定するのみ
【出力の安定性と属人化】	プロンプトの上手い個人に ノウハウが閉じ、出力がブレる	事前定義ファイルにより チーム全員がブレない構造化出力を維持
【所要時間（例：テーマ評価）】	手動設定と試行錯誤で 数日〜数週間	定義済みスキルの即時実行で 数分〜数時間
【データ処理（通話分析）】	1件ずつ文字起こしを 読んで手作業で転記	フォルダ内の28件のデータを 自動読み込みし即時生成


Low（低）：最速応答。単純なテキスト整形や軽微な変換用。
Medium（中）：標準レベル。
High / xHigh（高/超高）：ツールの複数回試行、ドキュメントの深い読み込み、複雑なロジック検証を自発的に行う。（※スキルファイル内で `effort: high` で設定）
Max（最大）：思考リソースと試行回数を最大解放（セッション限定）。
RPM
High
xHigh
スキルを思い通りに動かすための心臓部が『エフォートレベル』だ。AIがどれだけ深く思考し、自己検証を繰り返すかの『エンジンの出力』だよ。
Max（最大）、思考リソースと自己検証を繰りすかの『エンジンの出力』。
なるほど！複雑な分析スキルを動かす時は、自動的に高出力のギアに入るように設定しておいんですね。

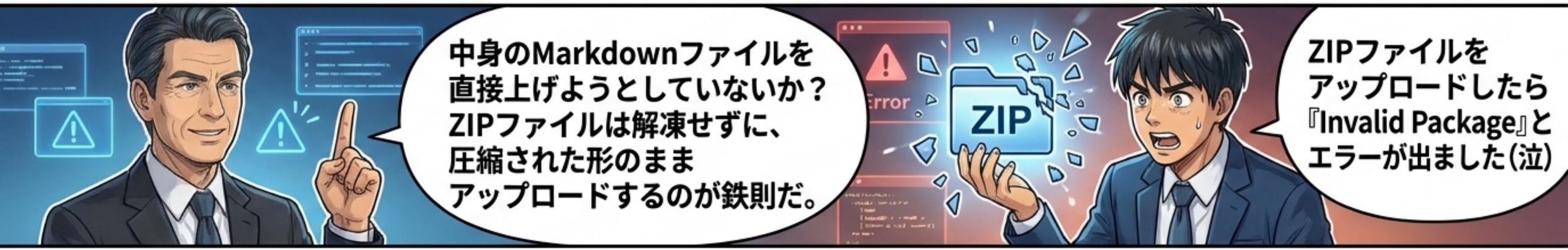

【強制上限】
組織の最大上限 (Organization Cap)。※全てに優先
【第1優先：環境変数】
`export CLAUDE_CODE_EFFORT_LEVEL=high`
（システム全体で強制適用。下位を上書き）
【第2優先：スキルのフロントマター】
`effort: xhigh`（スキル実行中のみ有効）
【第3優先：セッション設定】
`/effort high` または `--effort max`（進行中の会話内で指定）
【第4優先：設定ファイル】
`settings.effortLevel`（新規セッションの初期レベル）
【第5優先：モデルデフォルト】
`/effort auto`（上位指定がない場合の標準）
ただし注意しろ。
エフォートの設定には明確な
明確な『優先順位』の掟がある。
これを理解していないと、
いざという時にＡＩが
浅思考で止まってしまうぞ。
なるほど。
スキルのファイルで
『Ｈｉｇｈ』に設定していても、
ＰＣの環境変数で『Ｌｏｗ』に
環境変数が勝ってしたら
勝ってまうんですね。

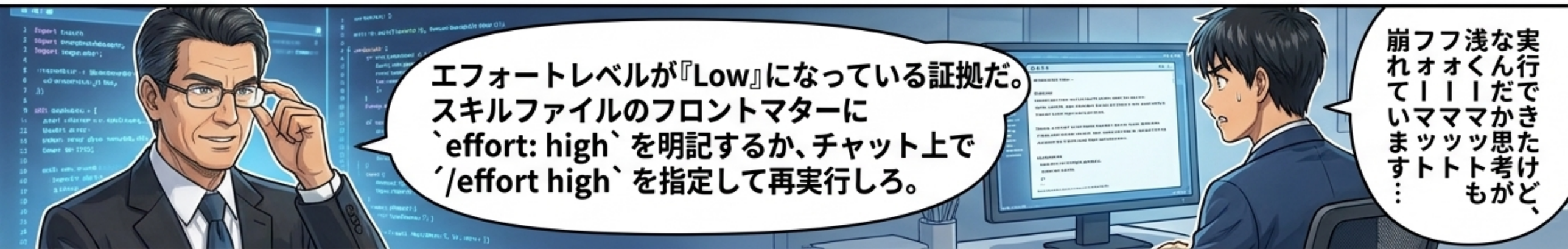

仕組みは分かりました！
では、実際に公式の
『科学的問題選択スキル
(scientific-problem-selection)』
導入してみます。
どうすればいいですか？
【個人アカウントの場合】
[設定] > [機能] > [スキル] ➡
「スキルをアップロード」
【組織管理（Team/Enterprise）の場合】
[管理者設定] > [機能] > [スキル] ➡
ON
➡「組織内でのスキル利用」をON
➡[組織スキルライブラリ]タブから追加
※超重要ポイント：
ダウンロードしたZIPファイル（例：
`scientific-problem-selection-v1.1.1.zip`）は
絶対に解凍せず、そのまま
アップロードすること！
組織プランの場合は、管理者が
管理者が一括アップロードすれば、
全メンバーの環境に即座に
即座に配布されるぞ。

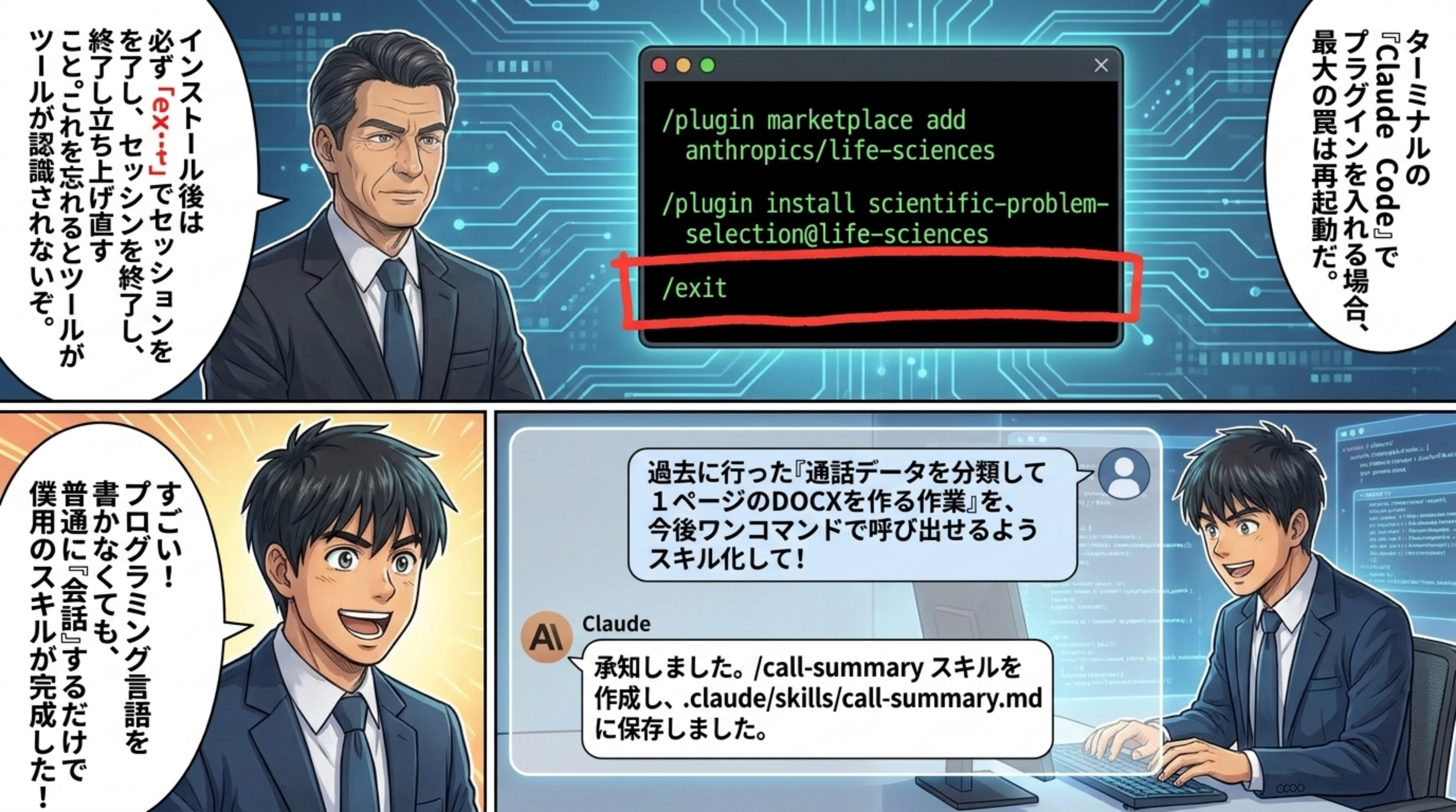

1. 直感ポンプ (Intuition Pump)
「仮に遅延が0秒になった場合、サイクル数はどう変化するか？」という鋭い問いの生成。
2. 仮定分析
仮定A
仮定B
結果1
結果2
結果1
結果2
3. パラメータ固定
コスト上限や制約の抽出。
4. 意思決定ツリー
[初期評価]
(S/N比 > 10)
全自動化へ
(S/N比 < 10)
高速分取へピボット
これこそがパッケージ化の力だ。
従来数週間かかっていた学術的かかっなように学術的フレームワークに沿ったた思考プロセスを、数分で展開・出力してくれる。
よし、実行プロンプトを入力…っと。クリークトにそれがすかは！
うわっ!?なんですかこの出力の速さと構造の美しさは！

スキル導入トラブルシューティング・ガイド








ターミナルの『Claude Code』でプラグインを入れる場合、最大の罠は再起動だ。
/plugin marketplace add anthropics/life-sciences
/plugin install scientific-problem-selection@life-sciences
/exit
インストール後は必ず「exit」でセッションを終了し、終了し立ち上げ直すこと。これを忘れるとツールがツールが認識されないぞ。
過去に行った『通話データを分類して１ページのDOCXを作る作業』を、今後ワンコマンドで呼び出せるようスキル化して！
Claude
承知しました。/call-summary スキルを作成し、.claude/skills/call-summary.md に保存しました。
すごい！プログラミング言語を書かなくても、普通に『会話』するだけで僕用のスキルが完成した！

スキル共有とスケジュール自動実行のワークフロー



[ストレージ]
`Calls/2026-Q1` フォルダ
（文字起こし＋分類体系ファイル）

（毎週月曜 午前9時に起動）



[実行エンジン]
`/schedule` コマンド

（エフォート: highで起動）



[共有スキル]
`/call-summary`（※チームで共有済み）

（全件読み込み・分類・抽出）



[自動出力]
`q1-2026-themes-brief.docx` が
指定フォルダに自動生成される。



毎週月曜の朝に出社したら、
すでに先週分のレポートが完成している…
夢のようなワークフローですね！

自作したスキルは自分だけのものにしておくのは勿体ない。
Coworkの『Share』ボタン一つでチーム全体のアセット（資産）になる。
さらに、`/schedule`コマンドで定期実行も可能だ。

「定型業務の箱から飛び出し、本質的な創造の領域へ」



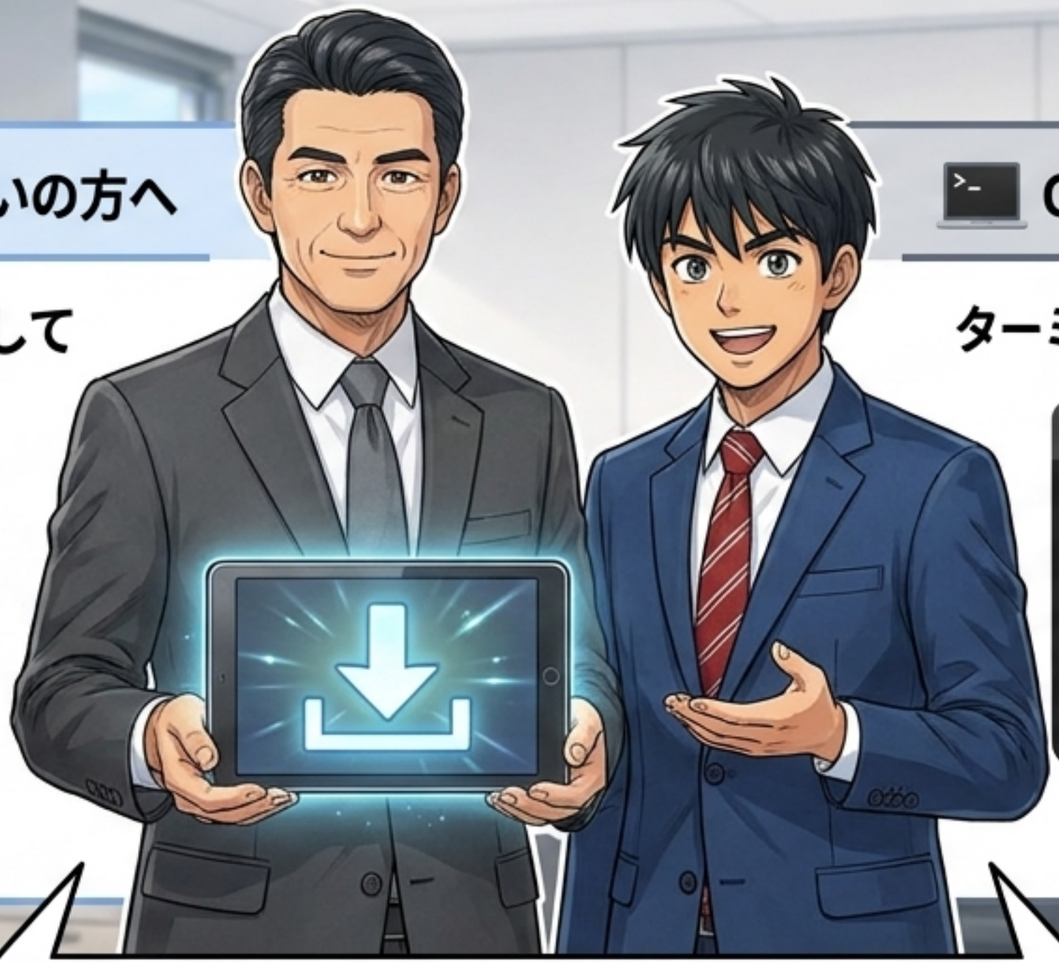
1. 作業速度と品質の飛躍的向上：プロンプトの手動入力やフォーマット修正の時間をゼロに。
2. 個人のノウハウからチームの資産へ：暗黙知だった分析手順が「スキル」として体系化され、組織全体の生産性を底上げする。
3. 人間が本来やるべき仕事への集中：下準備から解放され、「分析結果に基づく意思決定」「戦略立案」「創造的な対話」に全エネルギーを注ぐことができる。

ファーストアクション：今日、最初の魔法をかけてみよう

Claude.ai (Web) をお使いの方へ

公式サンプルをダウンロードしてアップロード！

1. 公式の `scientific-problem-selection-v1.1.1.zip`を入手。
2. 画面の [設定] > [機能] > [スキル] からZIPのままアップロード。



Claude Code をお使いの方へ

ターミナルから3ステップで導入完了!

```
/plugin marketplace add anthropics/life-sciences
/plugin install scientific-problem-selection@life-sciences
/exit
```

(※再起動を忘れずに！)

「定型作業の自動化は、最初の一歩を踏み出すことでその圧倒的な価値を実感できます。さあ、あなたの業務にも『スキル』を取り入れましょう！」