8層チェック表

自動化の現在地を塗ります。Day1 の最初と Day2 の最後の2回使います。

自動化の層を下から積み上げた8段です。Day1 の冒頭に現在地を1回塗り、Day2 の最後にもう一度塗ります。2回の差が2日間の成果です。

塗り方は3段階だけにしてください。細かく刻むと2回目の比較ができなくなります。

- 未: 置いていない
- 一部: 置いてあるが、定義どおりには効いていない
- 済: 効いていることを機械で確かめられる

「効いている」の判定は人の記憶ではなく、右の確認方法の欄に書いたコマンドか画面で決めます。D2 の core.md の語彙で言えば、[A] で守っているものは「一部」、[M] に移ったものが「済」です。

対象のハーネスは1人1式に絞ってください。D2 と smart3pm の両方を持っている方は、2日間で育てるほうを先に決めてから塗ります。

層	中身	成熟度	到達の目安	確認方法	Day 1	Day 2
1	非対話実行 (`claude -p`)	GA	人が画面を見ていない状態で1タスクが終了コード0で終わる	`claude -p "..." --output-format json` の終了コードと `.result` を見る。承認待ちで止まるなら未		
2	実行中の介入点 (hooks)	GA	PreToolUse / PostToolUse / Stop のどれかが、人の合図なしに作業を差し戻している	フックへ手で JSON を流して終了コード2が返る。`.claude/hooks/tests/` に回帰テストがある		
3	仕事の分割 (subagents / skills / plugins / MCP)	GA	観点ごと・規模ごとに別定義が走り分けている	`/tasks` にサブエージェント名とモデルと effort が出る		
4	承認の代わり (permission modes / sandbox)	GA	破壊的操作が deny とフックで止まり、止まったことが記録に残る	`permissions.deny` に巻き戻せない操作が並び、`/bin/rm` 経由もフックで止まる		
5	長時間化 (compaction)	GA	1タスクが文脈切れで終わらない。口頭の禁止事項がコンパクション後も効いている	`/compact` をまたいで制約が守られるか。守られないものは設定へ移す		

層	中身	成熟度	到達の目安	確認方法	Day 1	Day 2
6	並列化(Agent Teams / Workflows)	Agent Teams は experimental	複数の観点レビューが同時に走り、結果が1か所に集まる	同じ差分に3本のレビュアーを当てて、指摘の重なりと差が数えられる		
7	起動の自動化(CI / pipeline schedule)	GitLab 連携は beta	人が起動しなくてもパイプラインが動き、朝に結果が残っている	GitLab の Pipelines に schedule 起因の実行履歴がある		
8	プログラムからの呼び出し (Agent SDK)	GA	自社のツールから直接エージェントを呼べる	SDK 経由の実行ログがある		

成熟度の欄は 2026-09-10 時点の公式ドキュメントの表記です。GA は GitHub Actions 連携までを含み、GitLab CI/CD 連携は beta、Agent Teams は experimental、Routines は research preview と書かれています。Routines は claude.ai ログイン前提で、API キー運用では `/schedule` が出ません。スケジューラを GitLab 側に置くのはこのためです。

出典: https://code.claude.com/docs/en/headless / https://code.claude.com/docs/en/gitlab-ci-cd

塗り終えた後の記入欄

Day1 の記入が終わったら、次の2行だけ書いてください。Day2 の最後に同じ欄をもう一度書き、行が動いたかを見ます。

記入欄	Day1	Day2
「済」が付いた層の番号		
次に埋めると一番効く層と、その理由		

第7層の位置づけ

CI の導入は新機能の追加ではありません。いまは機械強制のタグが、エージェントの完了報告に載ることで担保されていると伺っています。第7層は、その担保をパイプラインの実行記録に置き直す作業です。第7層に印が付くまで、第1層から第6層の「済」はすべて自己申告の上に乗っています。

Givery