

Plusieurs sessions Claude Code sur le même dossier, sans collision

La structure de mon dossier de travail, les sept règles qui la font tenir, et un prompt qui en pose la base dans un dossier vide.

La structure

CLAUDE.md

Ce que toute session lit au démarrage : qui je suis, les conventions, où trouver le reste. Il reste court et renvoie vers les autres fichiers.

.claude/rules/

Une règle par sujet : le nommage, l'ordre des outils, git et les reprises. On en change une sans toucher aux autres.

projects/<projet>/NEXT-SESSION.md

Le fichier de reprise d'un projet en cours : l'état actuel, ce qui reste à faire dans l'ordre, les questions ouvertes. J'en ai près d'une centaine aujourd'hui.

projects/<projet>/goals/

Les longues tâches confiées en autonomie : ce qui est acquis, les garde-fous, les missions et le moment où s'arrêter.

decisions/log.md

Le journal des décisions. On y ajoute des lignes, on n'en réécrit jamais.

MEMORY.md (hors du dossier)

La mémoire automatique de Claude Code, qu'il range lui-même dans le dossier de l'utilisateur : un fait par fichier, et cet index d'une ligne par fait, chargé à chaque session. Rien à créer.

scripts/hooks/

Les contrôles qui se lancent seuls : au démarrage d'une session, avant une commande, après une écriture.

Les sept règles

1. Un projet, un fichier de reprise

Une session qui reprend un projet commence par lire son `NEXT-SESSION.md`. Si le sujet reste en cours à la fin, elle le met à jour. Deux sessions sur deux projets différents ne lisent jamais le même fichier.

2. Le fichier de reprise peut mentir

Il a pu vieillir pendant qu'une autre session travaillait. Avant de dire qu'une tâche est « en attente », la session vérifie dans l'outil lui-même.

3. Un contrôle git au démarrage

Un petit script regarde s'il reste un fichier `.git/index.lock`, un rebase en cours ou un autre processus git. S'il ne dit rien, la session peut travailler.

4. Chacun commite ses propres fichiers

Pendant le travail, une session commite avec `git commit -- <chemins>` et la liste de ce qu'elle a touché. Pas de `git add -A`, qui embarquerait le travail de la session d'à côté. Je ne l'autorise que dans la synchronisation de fin de session, qui ramasse tout ce qui reste.

5. Un garde-fou avant chaque push

J'ai deux comptes GitHub. Un script vérifie, avant un push, que le compte actif est le bon pour ce dépôt.

6. Un relais pour les sous-agents trop longs

Quand un sous-agent approche de 600 000 tokens de contexte, il finit son étape, écrit un fichier de reprise et s'arrête. Un agent neuf repart de ce fichier.

7. Un budget commun

Toutes les sessions consomment la même limite d'usage. Au-delà de cinq sous-agents pour une même tâche, la session demande mon accord avant de les lancer.

Le prompt pour créer le squelette

À coller dans Claude Code, dans un dossier vide ou presque. Il pose la base (règles 1 à 4). Les règles 5 à 7 dépendent de vos comptes et de vos agents, elles viennent ensuite.

Crée dans ce dossier une structure pour faire travailler plusieurs sessions Claude Code en parallèle, sans contenu métier :

1. Un `CLAUDE.md` court : qui je suis (à compléter), mes conventions, et la liste des fichiers de `.claude/rules/` à suivre.
2. `.claude/rules/git.md` : avant un commit, vérifier `.git/index.lock`, un rebase en cours et les processus git ; pendant le travail, ne commiter que ses propres fichiers avec `git commit -- <chemins>`, sans `git add -A`.
3. `.claude/rules/reprises.md` : une session qui reprend un projet lit d'abord `projects/<projet>/NEXT-SESSION.md`, le met à jour si le sujet reste en cours, et vérifie l'état réel avant de croire ce fichier.
4. `projects/exemple/NEXT-SESSION.md` avec trois titres vides : État actuel, À faire dans l'ordre, Questions ouvertes. Et un dossier `projects/exemple/goals/` vide.
5. `decisions/log.md` avec une ligne d'exemple, en ajout seul.
6. `scripts/hooks/git_preflight.mjs`, lancé au démarrage de chaque session par un hook `SessionStart` dans `.claude/settings.json` : il affiche un avertissement s'il trouve `.git/index.lock`, un rebase en cours ou un autre processus git, et ne dit rien sinon. Montre-moi l'arborescence créée, sans rien commiter.

Ce que ça ne règle pas

Deux sessions qui modifient le même fichier au même moment se gênent encore. Je les garde donc sur des projets différents, et je relis ce qu'elles produisent avant que quoi que ce soit parte.

Le format vient d'un post de Dan Saffar sur la structure qui fait travailler plusieurs sessions Claude ensemble. Le contenu de cette fiche vient de mon propre dossier de travail.

Clément Cathala, GTM Engineer. Fiche gratuite, à copier, adapter et partager. clementcathala.com