L'AUDIT DE TES SKILLS

Tes skills Claude, remises à niveau

Les modèles changent, tes skills, non. Ceux d'aujourd'hui sont beaucoup plus malins, et une consigne pensée pour un ancien modèle peut faire baisser la qualité de tes résultats. Voici les sept règles du guide officiel d'Anthropic, une skill qui fait l'audit pour toi, et les prompts pour réparer.

1 Ce que tu reçois

La skill d'audit

Elle trouve toutes tes skills, les lit une par une et te rend un rapport règle par règle, avec une page à ouvrir dans ton navigateur. Elle ne modifie rien sans ton accord.

Recevoir la skill

Les prompts pour réparer

Un prompt par règle, pour que l'IA t'aide à corriger ce que l'audit a trouvé. Plus un audit express, sans rien installer.

Voir les prompts

Faits avec Claude, ils marchent aussi avec ChatGPT, Codex et les autres outils qui lisent les skills : le format des skills est un standard ouvert, créé par Anthropic.

2 Les sept règles, dans la vie d'une skill

Une skill, c'est une fiche de consignes que Claude suit pour une tâche qui revient : tes devis, tes posts, tes rapports clients. Les règles suivent son parcours.

1

Qu'il la trouve

1 · Une description qui permet de la trouver

Au démarrage, Claude ne lit que le nom et la description de chaque skill. C'est l'étiquette sur le tiroir : ce que fait la skill, puis quand l'utiliser, à la troisième personne.

À éviter

« Je t'aide à écrire tes posts LinkedIn. »

À la place

« Rédige des posts LinkedIn dans mon style. À utiliser quand je demande un post ou une publication LinkedIn. »

2-3

Ce que tu lui dis

2 · Libre ou à la lettre : tu choisis

Pour chaque étape, demande-toi : qu'est-ce qui se passe s'il fait autrement ? Le mail qui accompagne un devis reste libre. Le calcul du total ne laisse aucune marge.

3 · L'objectif, pas le mode d'emploi

Les modèles récents sont très capables : si tu leur dictes chaque geste, tu les brides. Donne l'objectif, ce qui compte et les points où il ne faut pas se tromper. Retire les « IMPORTANT » et les majuscules, sauf là où une erreur coûte cher. Et dis à Claude quel modèle va lire ta skill : il dosera les détails.

4

Qu'il la lise

4 · Qu'il la lise en entier

Quand un fichier annexe renvoie vers un autre, Claude peut n'en lire que les cent premières lignes. Deux réflexes : chaque annexe appelée directement depuis la fiche principale, et un sommaire en tête de tout fichier de plus de cent lignes.

5-6

Qu'elle vérifie son travail

5 · La boucle

Vérifier, corriger, recommencer. Contre quelque chose de concret : ton guide de voix, ta charte, une liste de critères. « Vérifie que c'est bien » ne marche pas.

6 · La liste à cocher

Quand l'ordre compte, une liste que Claude recopie et coche, avec une consigne de retour : « si le total ne correspond pas, retourne à l'étape 3 ».

7

Qu'elle voyage

7 · Ce qu'il faut installer

Ta skill marche chez toi parce que ton ordinateur a déjà tout. À côté de chaque outil qu'elle utilise, écris la ligne pour l'installer. Et garde tes consignes dans le texte de la skill, pas dans les réglages d'un outil : elle passera de Claude à Codex sans rien perdre.

+

Qu'elle tienne dans le temps

Trois tests, et tes skills téléchargées

Avant d'écrire une skill, fais la tâche sans elle et note où Claude se trompe. Écris trois vraies demandes avec le résultat attendu, et relance-les à chaque modification ou changement de modèle.

Une skill téléchargée change avec son éditeur : mets-la à jour, puis audite-la comme les tiennes. À jour ne veut pas dire conforme. Si elle a un défaut, signale-le à l'éditeur ou fais ta propre copie : la prochaine mise à jour effacerait ta correction.

3 Lancer l'audit

Installer la skill

Claude Code : dézippe `auditer-skills.zip` dans `~/.claude/skills/` (pour toutes tes sessions) ou dans `.claude/skills/` de ton projet.

Codex et les autres outils : dézippe-la dans leur dossier de skills.

L'app Claude : importe le zip dans la section Skills des paramètres de Claude, puis joins à ta demande la skill à vérifier.

Les scripts de l'audit demandent Python 3.10 ou plus récent, et rien d'autre à installer. Testée avec Claude Opus et Sonnet. Avec Haiku, l'audit mesure juste, mais il juge moins bien les règles 1, 2, 5 et 6 : préfère Sonnet ou Opus.

Puis tape

Audite mes skills.

Elle commence par recopier sa liste à cocher, la remplit au fil de l'audit, puis te demande quelles corrections appliquer.

Reçois la skill sur **thomrichard.com/audit-skills**

4 Sans rien installer : l'audit express

Colle ce prompt dans Claude, ChatGPT ou un autre outil, avec le fichier `SKILL.md` de ta skill en pièce jointe.

L'audit express

```
Voici ma skill [nom] (fichier joint).
Vérifie-la avec ces sept règles et donne, pour chacune : OK, à
corriger ou non applicable, avec le passage concerné.
1. Description : elle dit ce que fait la skill puis quand l'utiliser,
à la troisième personne, jamais « je » ni « tu ».
2. Liberté : les étapes risquées (argent, envoi, suppression) suivent
une consigne exacte, les étapes créatives restent libres.
3. L'objectif, pas le mode d'emploi : pas de majuscules ni de
« IMPORTANT » répétés hors des étapes risquées, et la skill dit pour
quel modèle elle est écrite.
4. Lecture : chaque fichier annexe est appelé directement depuis
SKILL.md, et tout fichier de plus de 100 lignes commence par un
sommaire.
5. Boucle : le résultat est vérifié contre quelque chose de concret
(guide de voix, charte, critères), puis corrigé.
6. Liste à cocher : si l'ordre des étapes compte, la skill donne une
liste à recopier et cocher.
7. Installation : chaque outil utilisé a sa ligne d'installation à
côté.
Termine par la liste des corrections, de la plus importante à la
moins importante.
Ne modifie rien.
```

5 Réparer, une règle à la fois

Remplace ce qui est entre crochets. Chaque prompt se termine par « Ne modifie rien avant mon accord » : tu relis, puis tu choisis ce que tu gardes.

Règle 1 · Une description qui permet de la trouver

```
Réécris la description de la skill [nom] à la troisième personne :
ce qu'elle fait, puis quand l'utiliser, avec les mots que
j'emploierais pour la demander.
Montre-moi l'avant et l'après.
Ne modifie rien avant mon accord.
```

Règle 2 · Jusqu'où tu le laisses libre

```
Relis la skill [nom] étape par étape.
Pour chaque étape, dis-moi ce qui se passe si tu fais autrement que prévu, et classe-la : liberté haute, moyenne ou basse.
Signale les étapes où la consigne est trop stricte pour rien, et celles où elle ne l'est pas assez (argent, envoi, suppression, publication).
Ne modifie rien avant mon accord.
```

Règle 3 · L'objectif, pas le mode d'emploi

```
Relis la skill [nom] et repère les consignes écrites pour un ancien modèle : majuscules, « IMPORTANT », « TOUJOURS », répétitions, listes de cas à n'en plus finir.
Propose une version qui donne l'objectif, ce qui compte et les points où il ne faut pas se tromper, plutôt qu'un mode d'emploi geste par geste.
Garde la fermeté seulement là où une erreur coûterait cher.
Cette skill sera exécutée par [Opus / Sonnet / Haiku et Opus] : dose le niveau de détail pour ce modèle.
Ne modifie rien avant mon accord.
```

Règle 4 · Qu'il la lise en entier

```
Relis la skill [nom] et tous ses fichiers annexes.
1. Signale les fichiers qu'on n'atteint qu'en passant par un autre fichier, et propose le lien à ajouter directement dans SKILL.md.
2. Pour chaque fichier de plus de 100 lignes sans sommaire, propose un sommaire à placer en tête, qui reprend ses titres.
3. Si SKILL.md dépasse 500 lignes, propose ce qu'on déplace dans un fichier annexe.
Ne modifie rien avant mon accord.
```

Règle 5 · La boucle de vérification

```
Relis la skill [nom]. Là où la qualité du résultat compte, propose
une boucle de vérification :
contre quoi de concret vérifier (mon guide de voix, ma charte, une
liste de critères : demande-moi lequel si tu ne le trouves pas), quoi
corriger, et quand recommencer.
Remplace toute relecture vague du type « vérifie que c'est bien ».
Ne modifie rien avant mon accord.
```

Règle 6 · La liste à cocher

```
Relis la skill [nom]. Si l'ordre des étapes compte, propose une liste
à cocher que tu recopieras et cocheras à chaque utilisation,
avec une consigne de retour quand une étape échoue (« si …, retourne
à l'étape X »).
Si l'ordre ne compte pas, dis-le-moi et ne change rien.
Ne modifie rien avant mon accord.
```

Règle 7 · Ce qu'il faut installer

```
Pour chaque outil que la skill [nom] utilise, écris juste à côté la
ligne pour l'installer.
Puis dis-moi comment vérifier qu'elle marche sur un ordinateur où
rien n'est installé.
Ne modifie rien avant mon accord.
```

6 Garder tes skills à jour

Trois tests

À lancer avant de modifier une skill, puis à chaque fois que tu la modifies ou que le modèle change.

```
Propose-moi trois tests pour la skill [nom] : trois vraies demandes
que je pourrais lui faire, avec le résultat attendu pour chacune.
Puis lance-les et dis-moi lesquels passent, et pourquoi les autres
échouent.
```

Créer une nouvelle skill

```
Je veux une skill pour [la tâche].
Avant de l'écrire, fais la tâche sans skill sur cet exemple :
[exemple réel]. Puis dis-moi où tu t'es trompé ou ce que tu ne
pouvais pas deviner.
Propose ensuite trois tests, chacun avec le résultat attendu.
Puis écris la skill pour corriger ces erreurs-là, rien de plus. Elle
sera exécutée par [modèle] : assez guidée pour lui, sans sur-
expliquer.
Donne-lui une description à la troisième personne : ce qu'elle fait,
puis quand l'utiliser.
```

Pour aller plus loin

La chaîne YouTube

Des tutos concrets pour travailler avec Claude et l'IA au quotidien, sans jargon.

S'abonner sur YouTube

CLUB SMART

Tes skills, sans rester seul

Le Club SMART, c'est la communauté où l'on construit son système IA personnel avec Claude, puis où on le fait vivre semaine après semaine.

- Des parcours pas à pas pour installer et faire grandir ton système
- Des membres qui avancent sur les mêmes sujets que toi
- Un endroit pour partager tes avancées et poser tes questions

Découvrir le Club SMART