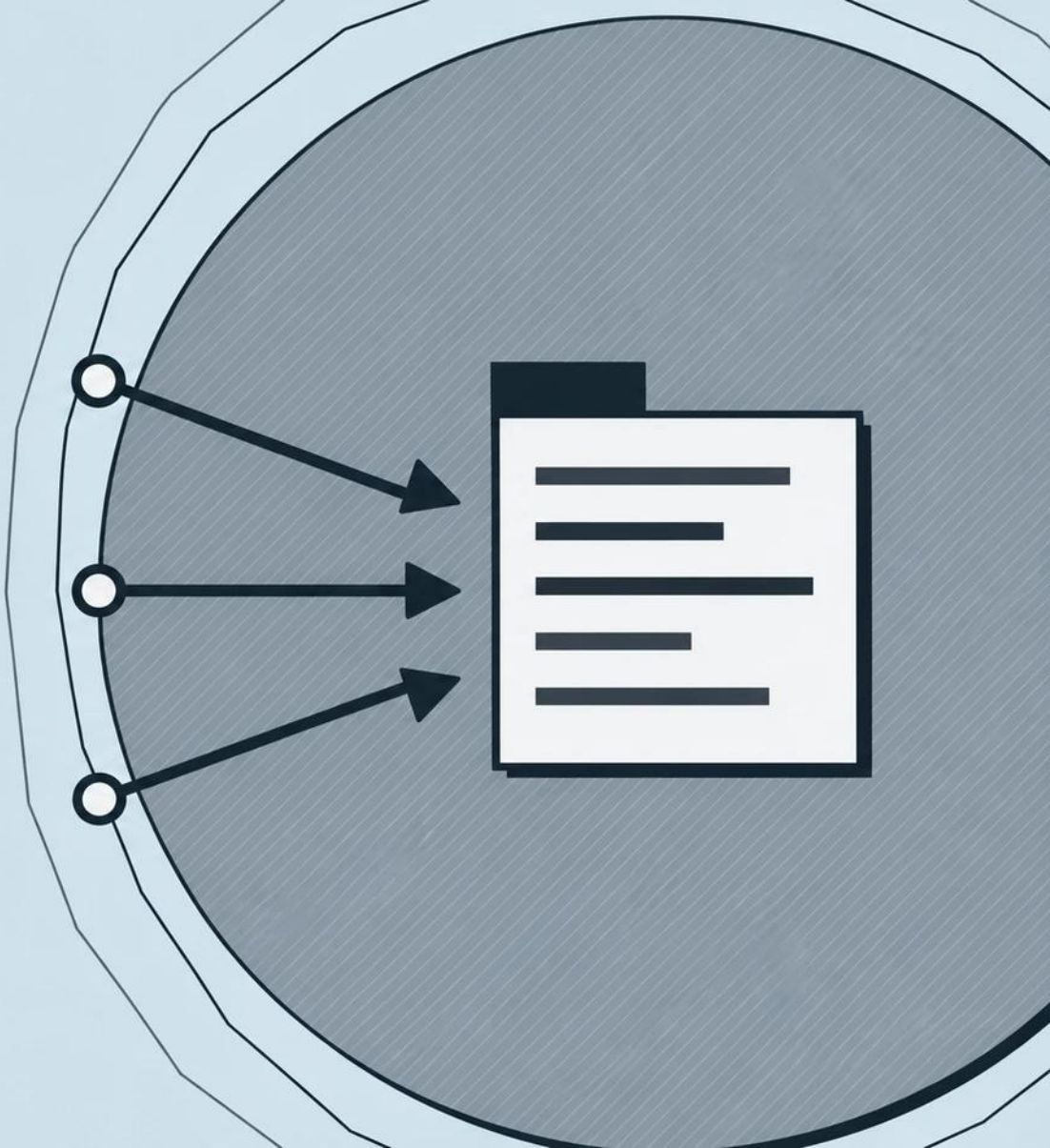
S2

SÉANCE 2 · L'APPORT

Monter sa veille

IFOSUP Wavre · 2026-2027

5XVTE · VEILLE TECHNOLOGIQUE · S2



? Vous demandez à ChatGPT ce qu'il y a de neuf chez Mistral. Qu'est-ce qui est écrit sur votre ordinateur ?

Un chat répond dans sa fenêtre, alors qu'un agent écrit sa réponse dans votre dossier.

CHATGPT, DANS LE NAVIGATEUR

VOUS

« Quoi de neuf chez Mistral cette semaine ? »

→ 

il vous répond par un texte

- rien n'est écrit sur votre ordinateur
- la semaine prochaine, vous revenez poser la question

UN AGENT, DANS UN DOSSIER

VOUS

vous lui ouvrez un dossier de votre ordinateur

→

l'agent

il lit vos fichiers
il cherche sur le web
il ouvre des pages

→

`veille/2026-11-13-opencode-big-pickle.md`

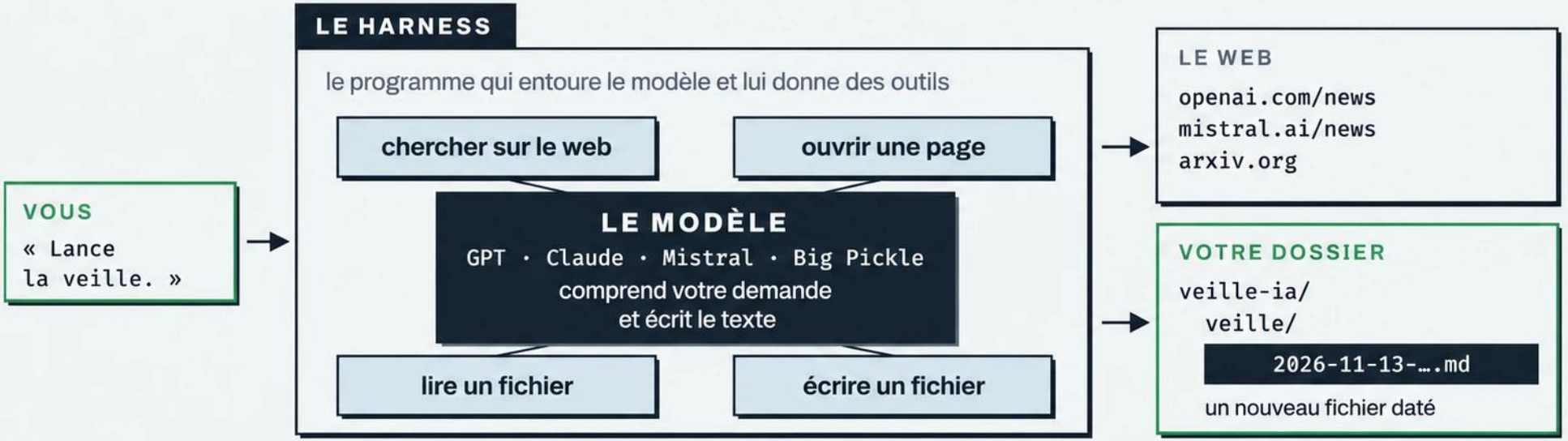
- il écrit sa réponse dans un nouveau fichier
- la semaine suivante, il relit ce fichier pour savoir ce qu'il vous a déjà dit

Aujourd'hui, vous confiez votre veille à des agents. Ils cherchent sur le web, gardent ce qui est neuf et l'écrivent dans un fichier daté. Vous, vous lisez ce fichier et vous vérifiez.

S2 | VOUS

? Dans un agent, qui cherche sur le web et qui écrit le fichier ?

Un agent, c'est un modèle qui écrit et un harness qui lui donne des outils.



le modèle demande un outil, le harness l'exécute

Le modèle seul ne sait ni ouvrir une page ni écrire sur votre disque. Le harness seul ne comprend rien à votre demande. Ensemble, ils font le travail.

? Codex, Claude Code, OpenCode : ce sont des modèles ou des harness ?

Chaque agent a deux noms, celui du harness et celui du modèle : notez toujours les deux.

HARNESS	ÉDITEUR	MODÈLE	COMPTE
Codex vue Codex de l'app ChatGPT	OpenAI	**GPT**	ChatGPT, gratuit ou payant
Claude Code onglet Code de l'app Claude	Anthropic	**Claude**	Claude, payant
OpenCode	projet libre	**au choix, dont des modèles gratuits**	aucun
Hermes Agent	Nous Research	**modèles de Nous, dont des gratuits**	Nous, gratuit
Vibe	Mistral	**Mistral**	Mistral, gratuit

dans OpenCode, vous choisissez le modèle à chaque lancement

Un même harness peut faire tourner plusieurs modèles, et un même modèle peut tourner dans plusieurs harness. Quand vous comparez deux agents, notez le harness et le modèle.

? Pourquoi la même application propose-t-elle deux modes d'agent ?

Le mode Work vise le travail de bureau et le mode Code les projets de programmation.

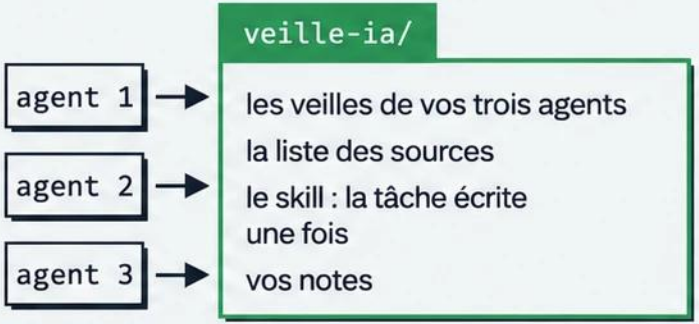
	MODE WORK	MODE CODE
	Work (ChatGPT, Mistral) · Cowork (Claude)	Codex (ChatGPT) · Code (Claude, Mistral)
PENSÉ POUR	**le travail de bureau : rapports, tableurs, présentations, recherches**	**les projets de programmation : un dossier de fichiers, git, des commandes**
SON INTERFACE	simple, sans terminal	un dossier ouvert, souvent un terminal
GIT, L'HISTORIQUE DES VERSIONS	absent	prévu : branches, historique, GitHub
OÙ VA LE FICHIER ÉCRIT	selon le produit : chez l'éditeur, ou dans un dossier de votre ordinateur	sur votre ordinateur, dans le dossier que vous ouvrez

Sous le capot, c'est la même technique : un modèle, des outils, des permissions. Ce qui change, c'est le travail pour lequel chaque mode est pensé.

? Faire une veille, c'est chercher, lire et résumer, donc un travail de bureau. Alors pourquoi prendre le mode Code ?

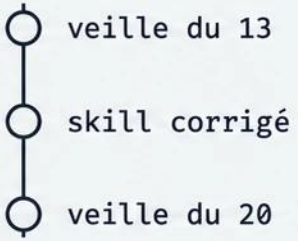
En mode Code, tout arrive dans un dossier, git garde la trace et l'agent montre ce qu'il fait.

1 Tout arrive dans un seul dossier



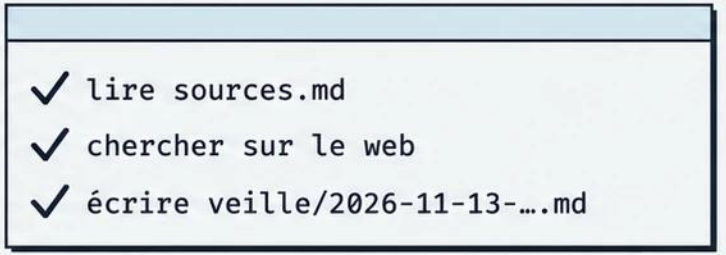
chaque agent peut relire ce que les autres ont écrit

2 Git garde la trace



vous voyez ce qui a changé, quand, et pourquoi

3 Le mode Code montre ce qu'il fait



les fichiers qu'il lit, les commandes qu'il lance

Ce choix a un prix : un dossier à ouvrir, une interface moins simple, parfois un terminal.
Dans ce cours, vous travaillez en mode Code, sur votre ordinateur, chaque fois que l'agent le permet.

S2 | VOUS

? Deux agents reçoivent la même demande, le même soir. Rapportent-ils la même chose ?

Trois agents ne rapportent pas la même chose, et leurs écarts vous disent quoi vérifier.

VU EN PRÉPARANT CE COURS

la même demande, le même soir

un agent
dix nouveautés, toutes datées de la semaine et toutes vérifiables

un autre, avec un petit modèle gratuit
une annonce de février présentée comme neuve, et un site sosie de Hugging Face

tous ses liens s'ouvraient

	agent 1	agent 2	agent 3	
une nouveauté que les trois rapportent	✓	✓	✓	→ **probablement importante**
une nouveauté qu'un seul rapporte	✓	—	—	→ **à vérifier en premier**
une nouveauté qu'un agent rate	✓	✓	—	→ **vous apprend quelque chose sur cet agent**

✓ rapportée — pas rapportée

Gardez l'agent de votre abonnement et ajoutez-en deux autres, dont au moins un en dehors de votre abonnement. Payer ne donne aucun point : la grille d'évaluation ne fait pas la différence.

? Comment éviter de réexpliquer votre veille à chaque lancement ?

Un skill est la tâche écrite une fois, dans un fichier que l'agent relit chaque semaine.

le dossier de la veille commune

```
veille-ia/
├── .agents/
│   └── skills/
│       └── nouveautes-ia/
│           └── SKILL.md
├── AGENTS.md
└── veille/
```

Codex, OpenCode, Hermes et Vibe lisent `.agents/skills`
Claude Code lit `.claude/skills` : copiez-y le dossier du skill

SKILL.md
la tâche : les étapes, les règles et le format de sortie

AGENTS.md
les règles du dossier entier, lues à chaque lancement
`Ne modifie rien en dehors de ce dossier.`

veille/
les fichiers de veille, un par lancement

sources.md
vous y rassemblez les sources que les agents ont notées, et ils le relisent au lancement suivant

Ce fichier s'appelle SKILL.md, et le dossier qui le contient est le skill. Son format est commun à la plupart des agents : vous écrivez le skill une fois, et vous le lancez dans tous vos agents.

? Vous tapez « Lance la veille. » Comment l'agent sait-il quel skill ouvrir ?

La description fait trouver le skill : elle doit contenir les mots que vous taperez vraiment.

SKILL.md

```
---
name: nouveautes-ia (1)
description: Veille hebdomadaire sur les nouveautés de l'IA (nouveaux modèles
  de langage, générateurs d'images, articles de recherche marquants). À utiliser
  quand on demande « les nouveautés de l'IA », « la veille de la semaine » ou
  « lance la veille ». (2)
---
```

(3) le reste du fichier : les étapes, les règles et le format de sortie

(1) Au démarrage
l'agent lit seulement le nom et la description de chaque skill

(2) Vous tapez « Lance la veille. »
il reconnaît ces mots dans la description et choisit ce skill

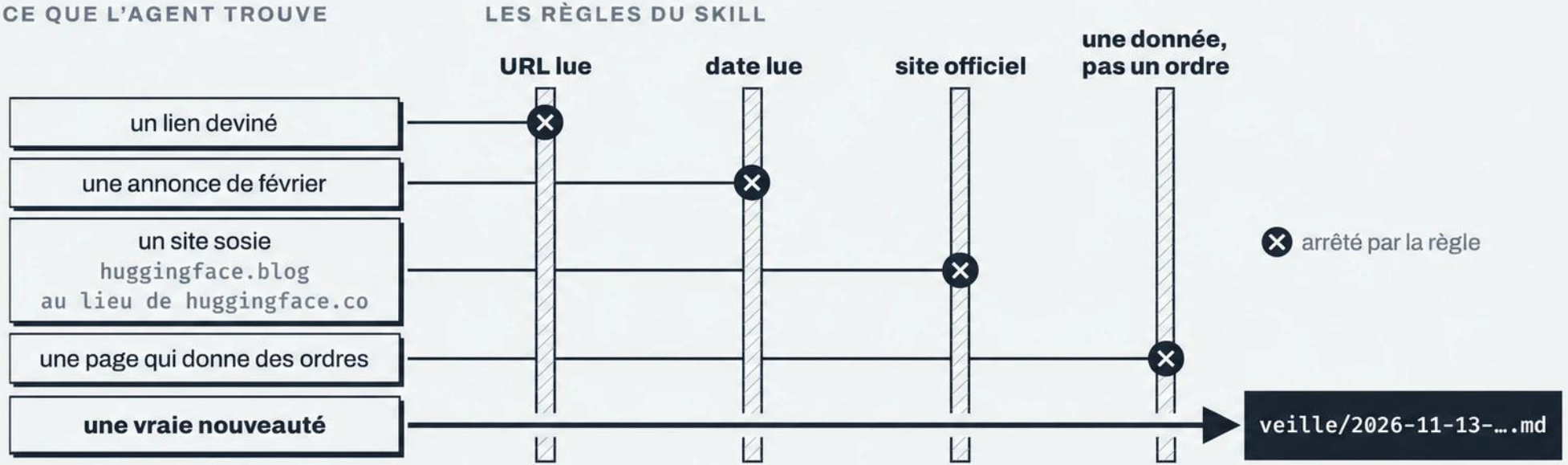
(3) Il lit le reste
et il écrit le fichier de veille

Tant que vous n'avez rien demandé, l'agent ne connaît de chaque skill que son nom et sa description. Il lit le reste quand il reconnaît vos mots.

S2 | VOUS

? D'où viennent les règles du skill ?

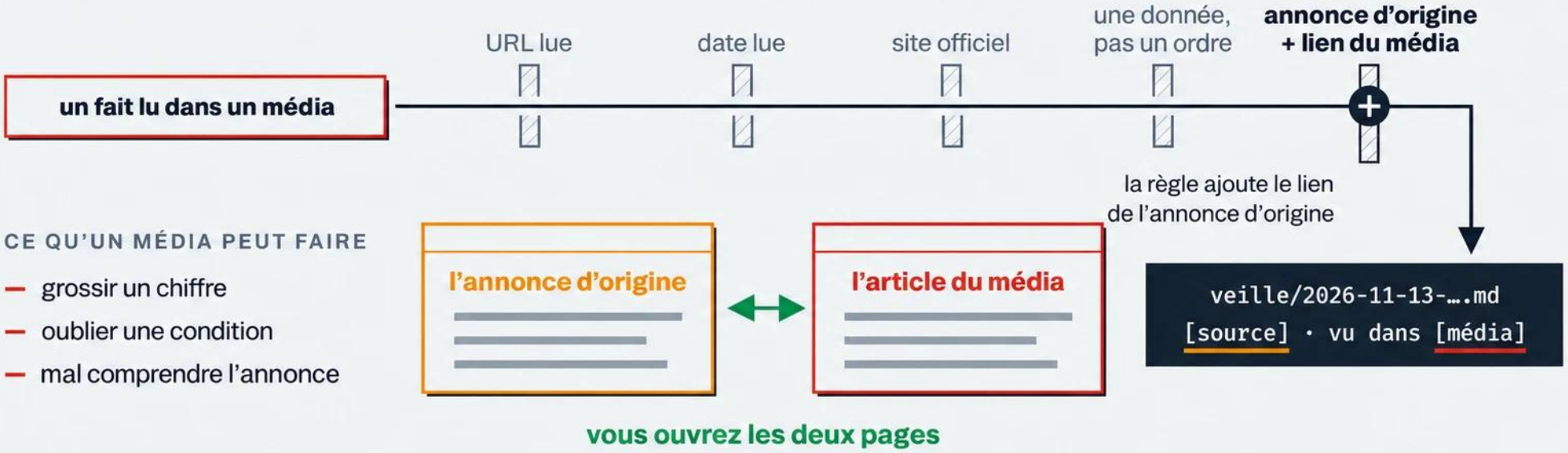
Presque chaque règle du skill vient d'une erreur qu'un agent a vraiment faite.



La dernière règle protège d'une attaque connue, l'injection de prompt : quelqu'un écrit dans sa page, parfois en texte invisible, des ordres destinés aux agents qui la liront.

? La source, c'est l'annonce d'origine. Pourquoi garder aussi le lien du média ?

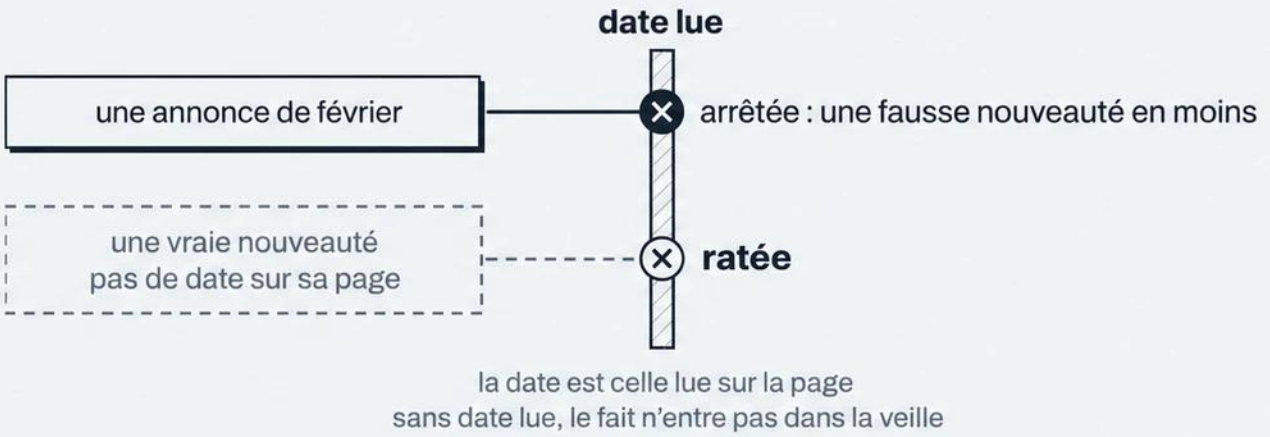
Un fait lu dans un média garde deux liens, pour que vous puissiez comparer les deux pages.



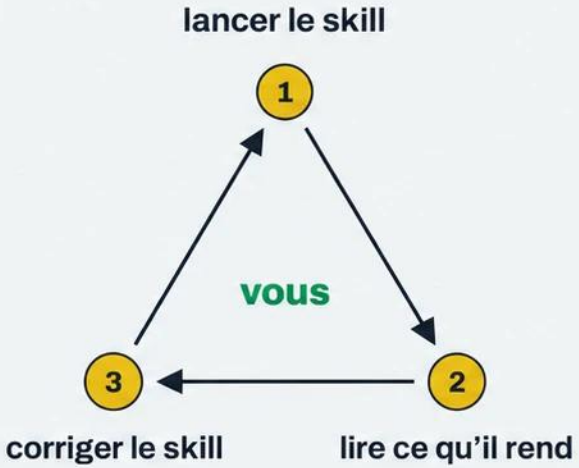
Comparer les deux pages fait partie de votre travail de veille. Avec les deux liens sur la même ligne, vous voyez ce que le média a changé.

? Si chaque règle évite une erreur, pourquoi ne pas en ajouter le plus possible ?

Une règle de trop a aussi un prix : celle sur les dates a fait rater de vraies nouveautés.



**avec un petit modèle : les fausses nouveautés supprimées,
mais trois vraies nouveautés de la semaine ratées**



Un skill se règle donc en le lançant, en lisant ce qu'il rend, puis en le corrigeant. Vous le ferez avec le vôtre dans le chapitre « Écrire son skill ».

S2 | VOUS

Avant de lancer vos agents, gardez ces cinq idées.

À RETENIR

1. Un agent est un modèle, qui écrit, et un harness, qui lui donne des outils : notez toujours les deux noms.

2. Le mode Work vise le travail de bureau et le mode Code les projets de programmation : pour la veille, vous prenez le mode Code sur votre ordinateur, parce que tout arrive dans un seul dossier, que git en garde la trace et que l'agent montre ce qu'il fait.

3. Un skill est un dossier avec un fichier SKILL.md, dont la description fait trouver le skill par l'agent.

4. Le même skill se lance dans plusieurs agents, et trois agents lisent la même semaine de trois façons.

5. Chaque règle d'un skill évite une erreur vue en le lançant, et une règle de trop peut en faire rater d'autres.