Jumeau numérique · Sous-section

Lexique IA

Glossaire des termes essentiels : token, MCP, API, provider, modèle, agent, ZDR, Workspace, opencode.json, skills, Obsidian Sync, et plus.

Se repérer dans le monde de l'IA

L'IA a son propre vocabulaire. Cette page est un glossaire de référence : reviens-y chaque fois que tu rencontres un terme que tu ne connais pas.

Les fondamentaux

Interface
OpenCode Desktop, Claude.ai, ChatGPT...
Ce que tu utilises pour parler au modèle
Provider
OpenRouter, Anthropic, OpenAI, Alibaba...
Qui fait tourner le modèle sur ses serveurs
Modèle (LLM)
DeepSeek V4 Flash, GLM 5.2, GPT 5.6, Qwen...
Le "cerveau" — l'intelligence elle-même

Modèle (LLM)

Un **Large Language Model** est un programme entraîné sur des milliards de textes pour comprendre et générer du langage. C'est le "cerveau" : l'intelligence elle-même. Exemples dans la stack du bootcamp : DeepSeek V4 Flash (open source, modèle par défaut), GLM 5.2 (Zhipu AI, pour les tâches complexes), Kimi 2.6 (Moonshot AI), Qwen (Alibaba, multilingue), GPT 5.6 (OpenAI, via ChatGPT Plus).

Un modèle ne fait rien tout seul. Il a besoin d'une interface pour recevoir tes messages et d'un provider pour faire tourner le calcul.

Provider

Le **provider** est l'entreprise qui fait tourner le modèle sur ses serveurs. C'est l'intermédiaire entre toi et le modèle. Exemples : OpenRouter (agrégateur recommandé dans le bootcamp), Anthropic, OpenAI, Alibaba.

Un même modèle peut être accessible via plusieurs providers. Qwen peut tourner chez OpenRouter, chez Alibaba, ou sur ta propre machine. Le prix, la vitesse et la confidentialité changent selon le provider.

Interface

L'**interface** est ce que tu utilises pour parler au modèle. Exemples : OpenCode Desktop (l'interface principale du bootcamp), Claude.ai (web), ChatGPT (web/app).

Une interface peut se connecter à différents providers et modèles. OpenCode Desktop peut utiliser DeepSeek, GPT, Qwen — il suffit de connecter le bon provider.

Token

Un **token** est l'unité de mesure du texte pour un modèle IA. Un token représente environ 3/4 d'un mot en anglais, un peu moins en français.

Quand on parle de "200K tokens de contexte", ça veut dire que le modèle peut traiter environ 150 000 mots d'un coup : l'équivalent d'un livre de 500 pages.

Les tokens servent aussi à la facturation : les providers facturent au nombre de tokens traités (entrée + sortie).

Fenêtre de contexte (Context Window)

La **fenêtre de contexte** est la quantité maximale de texte que le modèle peut "voir" en même temps. C'est sa mémoire de travail.

- La plupart des modèles : 128K-200K tokens
- Gemini : jusqu'à 1M tokens

Si tu dépasses la fenêtre de contexte, le modèle "oublie" le début de la conversation. C'est pour ça que les agents comme OpenCode gèrent le contexte intelligemment : ils envoient uniquement ce qui est pertinent.

Prompt

Un **prompt** est simplement le texte que tu envoies au modèle. Ça peut être une question, une instruction, un document à analyser. Un bon prompt est précis, contextuel et structuré. La qualité de la réponse dépend directement de la qualité du prompt.

Terminal

Le **terminal** est l'application qui permet de taper des commandes texte pour contrôler ton ordinateur. C'est la fenêtre noire (ou colorée) où tu tapes des instructions au lieu de cliquer avec la souris.

Exemples de terminaux : Ghostty, iTerm2, ou le Terminal par défaut de ton Mac/PC. OpenCode Desktop a son propre terminal intégré, donc la plupart des participants n'ont pas besoin d'en installer un.

CLI (Command Line Interface)

Une **CLI** (interface en ligne de commande) est un programme qu'on utilise en tapant des commandes dans le terminal. OpenCode peut s'utiliser en mode CLI, en plus de l'interface Desktop. GOG (pour Google Workspace) est une CLI.

Les concepts agentiques

Chatbot classique
Répond à des questions
Génère du texte uniquement
Pas d'accès à tes fichiers
Pas d'actions concrètes
Agent IA
Exécute des actions
Lit et écrit des fichiers
Appelle des outils (MCP)
Enchaîne des étapes en autonomie

Agent

Un **agent** est un modèle IA qui peut exécuter des actions, pas seulement répondre à des questions. Il peut lire des fichiers, écrire du code, naviguer sur le web, appeler des APIs, et enchaîner ces actions en autonomie pour accomplir une tâche complexe.

OpenCode Desktop transforme un modèle en agent : il lui donne accès à ton système de fichiers, à tes outils, à tes commandes.

Tool Calling (appel d'outils)

Le **tool calling** est la capacité d'un modèle à utiliser des outils externes. Au lieu de juste générer du texte, le modèle peut décider d'appeler un outil (lire un fichier, faire une recherche web, interroger une base de données) et utiliser le résultat pour continuer son raisonnement.

C'est ce qui fait la différence entre un chatbot et un agent.

MCP (Model Context Protocol)

Le **MCP** est un standard ouvert créé par Anthropic pour connecter des outils à un modèle IA. C'est comme un port USB universel pour l'IA.

Un serveur MCP expose des capacités (lire une base de données, scraper une page web, générer une image) et n'importe quelle interface compatible (OpenCode Desktop, Claude Desktop) peut les utiliser.

OpenCode Desktop
Interface compatible MCP
se connecte via MCP à
Firecrawl
Web
n8n
Automations
NocoDB
Bases de données
QMD
Vault search

Exemples de MCP qu'on utilise dans le bootcamp : Firecrawl (web), n8n (automatisations), NocoDB (bases de données), QMD (recherche dans le coffre).

Skill (commande slash)

Un **skill** (ou commande slash) est un fichier Markdown qui contient des instructions pour l'IA. Quand tu tapes `/newsletter` dans OpenCode, il lit le fichier `Second Cerveau/4 TOOLS/skills/newsletter.md` et exécute les instructions qu'il contient.

C'est le mécanisme qu'on utilise pour créer des agents personnalisés. Le fichier contient le contexte, la méthode, les étapes : tout ce que l'IA a besoin de savoir pour exécuter le process.

À ne pas confondre avec les commandes systèmes d'OpenCode (comme `/model` ou `/sync`) qui sont des fonctions intégrées à l'interface.

Fork

Un **fork** est une copie d'un projet open-source qu'on modifie pour ses propres besoins. OpenCode est un fork de Claude Code : il est parti du code de Claude Code et a évolué indépendamment.

Infrastructure et technique

API (Application Programming Interface)

Une **API** est une interface qui permet à des programmes de communiquer entre eux. Quand OpenCode envoie ta question à DeepSeek via OpenRouter, il utilise l'API d'OpenRouter.

L'API est l'alternative aux interfaces web (chatgpt.com, claude.ai). Elle est plus flexible, plus puissante, et permet l'automatisation — mais elle nécessite une clé API et se facture à l'usage.

Clé API (API Key)

Une **clé API** est un mot de passe unique qui t'identifie auprès d'un provider. Quand tu connectes OpenRouter dans OpenCode, tu fournis ta clé API OpenRouter.

Règle absolue : ne commite jamais une clé API dans un repo Git, et ne la partage jamais. Si quelqu'un l'obtient, il peut utiliser ton compte et tu seras facturé. Stocke tes clés dans Bitwarden.

ZDR (Zero Data Retention)

Le **ZDR** signifie que le provider ne conserve aucune donnée après traitement. Ta requête arrive, la réponse est générée, tout est supprimé immédiatement. Rien n'est stocké, rien n'est loggé.

OpenRouter propose le ZDR activable en quelques clics. C'est la configuration recommandée dans le bootcamp. Voir la leçon Zoom sur la sécurité pour le détail.

opencode.json

Le fichier **opencode.json** est le fichier de configuration d'OpenCode, placé à la racine de ton Workspace. Il définit les providers connectés, les MCPs actifs, les permissions (ce que l'IA peut ou ne peut pas faire), et les options d'affichage. C'est l'équivalent d'un fichier de configuration de projet.

AGENTS.md

Le fichier **AGENTS.md** (ou `claude.md` dans certains contextes) est un fichier d'instructions permanent que l'IA lit à chaque session. Il contient ton identité, tes préférences, le contexte de ton projet, et les règles que tu veux qu'elle respecte. C'est la mémoire de travail persistante entre les sessions.

Dans le bootcamp, ce rôle est rempli par ton fichier `Moi.md` et les contextes dans ton coffre.

GitHub / GitLab

GitHub et **GitLab** sont des plateformes de stockage et de versioning de fichiers. Elles gardent un historique complet de toutes les modifications : tu peux revenir en arrière à tout moment.

GitHub appartient à Microsoft (serveurs aux USA). GitLab peut être auto-hébergé en Europe.

Repository (Repo)

Un **repository** (ou "repo") est un projet sur GitHub/GitLab. C'est un dossier versionné : chaque modification est enregistrée avec un horodatage et un message.

Inférence

L'**inférence** est le moment où le modèle génère une réponse. "Faire de l'inférence" = faire tourner le modèle pour produire du texte. Les providers d'inférence (OpenRouter, DeepInfra) sont des entreprises qui font tourner des modèles sur leurs serveurs sans les entraîner.

n8n

n8n est un outil d'automatisation open-source (comme Zapier ou Make, mais auto-hébergé). Il permet de créer des workflows visuels qui connectent des services entre eux. On l'installe sur ton propre serveur en semaine 3 du bootcamp.

Ollama

Ollama est un outil pour faire tourner des modèles IA en local sur ta machine. Au lieu d'envoyer tes données à un serveur distant, le modèle tourne directement sur ton ordinateur. Avantage : confidentialité totale. Inconvénient : nécessite du matériel puissant pour des performances correctes.

SSH (Secure Shell)

SSH est un protocole sécurisé pour se connecter à un serveur distant depuis ton terminal. C'est comme une télécommande chiffrée pour ton serveur. On l'utilise en semaine 3 pour accéder au serveur IA.

Webhook

Un **webhook** est une URL qu'un service appelle automatiquement quand un événement se produit. Exemple : "Quand quelqu'un paie sur mon site, appeler cette URL, puis créer son compte". Les webhooks sont le mécanisme de base des automatisations n8n.

Open-source vs propriétaire

Un modèle **open-source** (Qwen, DeepSeek, Llama) a son code et ses poids publiés : n'importe qui peut le télécharger, le modifier, le faire tourner. Un modèle **propriétaire** (GPT 5.6, Claude) est accessible uniquement via l'API du créateur.

L'open-source permet le self-hosting et offre plus de contrôle sur la confidentialité. DeepSeek V4 Flash est open-source et constitue le modèle par défaut recommandé dans le bootcamp.

Le coffre et l'infrastructure

Workspace

Le **Workspace** est le dossier racine depuis lequel tu lances OpenCode. Dans le bootcamp, c'est le dossier que tu télécharges au départ : il contient un sous-dossier `Second Cerveau/` (ton coffre Obsidian), un fichier `opencode.json`, et les fichiers de configuration de l'agent.

Vault (Coffre) / Second Cerveau

Un **vault** (ou "coffre") est le dossier principal d'Obsidian qui contient toutes tes notes. C'est ton second cerveau : un ensemble de fichiers Markdown liés entre eux par des wikilinks. Dans le bootcamp, il s'appelle `Second Cerveau/` et vit à l'intérieur du Workspace.

Obsidian Sync

Obsidian Sync est le service de synchronisation payant d'Obsidian. Il chiffre tes notes de bout en bout et les synchronise entre tous tes appareils (Mac, iPhone, iPad). C'est la solution recommandée dans le bootcamp pour synchroniser ton coffre (8$/mois avec le plan annuel). Les données passent par les serveurs d'Obsidian mais restent chiffrées.

Markdown (.md)

Le **Markdown** est un format de fichier texte simple utilisé par Obsidian. Un fichier `.md` est du texte brut avec des conventions de formatage (`# titre`, `**gras**`, `- liste`). C'est le format natif d'Obsidian et le format que l'IA lit le mieux. Il est compatible partout et n'est pas propriétaire.

Wikilink

Un **wikilink** (`[[Nom de la note]]`) est un lien interne entre deux notes dans Obsidian. C'est ce qui crée le réseau de connexions de ton second cerveau.

IPCRA

IPCRA est le système d'organisation du coffre : **I**nbox, **P**rojets, **C**asquettes, **R**essources, **A**rchives. Chaque note va dans une de ces catégories. C'est une adaptation du système PARA de Tiago Forte.

Phase de vie

Une **phase de vie** est une période définie avec une intention claire. Exemples : "Lancer mon business", "Apprendre le machine learning", "Déménagement à Hong Kong". Les phases de vie structurent ton journaling et ta planification dans le coffre.

- [ ] J'ai compris la différence entre modèle, provider et interface
- [ ] Je sais ce qu'est un token et une fenêtre de contexte
- [ ] Je comprends ce que fait un agent vs un chatbot
- [ ] Je sais ce que signifie MCP, API, CLI, ZDR, SSH, webhook
- [ ] Je sais ce que sont le Workspace, le coffre, et opencode.json