

Vibe Coding: Formation Starter

Une journée pour apprendre à coder avec l'assistance de l'IA

Philippe Pary & Thomas Foutrein

2025

Objectifs de la formation

- **Coder** avec une assistance IA (Cursor)
- **Améliorer** du code existant
- **Intégrer** l'IA dans votre workflow
- **Personnaliser** l'outil à vos besoins
- **Formuler** des prompts efficaces
- **Développer** un projet complet en "Vibe Coding"

Tour de table

- **Qui êtes-vous ?** (Nom, fonction)
- **Votre expérience avec l'IA ?** (Déjà vibe codé ?)
- **Votre vision du vibe coding ?**
- **Vos attentes pour aujourd'hui ?**

Le Vibe Coding, c'est quoi ?

Une technique de programmation assistée par LLM où le **prompt** est au cœur du développement.

- **Plus qu'une simple auto-complétion.**
- Le développeur devient un **architecte** et un **guide** pour l'IA.
- **Transformation du métier** : plus de relecture, de conception et de tests.

Concept popularisé par Andrej Karpathy (co-fondateur d'OpenAI) en février 2025.

Comment ça marche ? (Les LLMs)

1. **Pré-entraînement** sur des milliards de lignes de code et de texte.
2. **Fine-tuning** sur des tâches spécifiques (génération de code, etc.).
3. **RLHF** (Reinforcement Learning from Human Feedback) pour affiner la qualité.

Processus de génération :

- **Tokenisation** du prompt.
- **Analyse du contexte** (code, fichiers ouverts).
- **Génération auto-régressive** (token par token).

⚠ **Limites** : Hallucinations, connaissance limitée, pas de test direct du code. Le développeur reste indispensable !



Les Outils du Vibe Coding

Outil	Type	Avantages	Inconvénients
Cursor	Éditeur dédié	✓ Optimisé, personnalisable	⚠ App séparée
gemini-cli	Outil CLI	✓ Scriptable, gratuit	⚠ Pas de GUI
Replit	En ligne	✓ Environnement complet	⚠ Connexion requise
Continue	Extension VSCode	✓ Open source, intégré	⚠ Moins stable
Copilot	Extension multi-IDE	✓ Mature, très répandu	⚠ Moins orienté "vibe"

Workflow : Deux approches

1. Itératif

On part d'un prompt initial, puis on ajoute les fonctionnalités une par une.

-  Meilleure maîtrise de l'évolution.
-  Risque de "partir en vrille". Pensez à commiter souvent !

2. One Big Happy Prompt

On prépare un cahier des charges très détaillé en amont.

-  Projet structuré, moins de régressions.
-  Relecture fastidieuse, temps de développement potentiellement plus long.

L'Art du Prompt Engineering

La qualité de votre prompt détermine la qualité du code généré.

Structure d'un prompt efficace :

1. **Contexte** : Objectif du projet, contraintes.
2. **Rôle** : "Tu es un développeur expert en Python..."
3. **Tâche** : "Développe une fonction qui..."
4. **Format de sortie** : "Le code doit être commenté, utilise des docstrings..."
5. **Exemples** : Données d'entrée et de sortie attendues.

Astuce : Utilisez les `rules` de Cursor pour définir un contexte permanent pour vos projets.

Pièges à éviter

- **Ne pas commiter assez souvent.** L'IA peut détruire votre travail. `git commit -m "checkpoint"` est votre ami.
- **Faire une confiance aveugle.** Relisez systématiquement le code généré.
- **Ignorer la sécurité.** Les LLMs peuvent introduire des vulnérabilités.
- **Utiliser des prompts vagues.** Soyez précis et technique.
- **Négliger les tests.** La couverture de tests est votre filet de sécurité.

À vous de jouer ! (Exercices)

1. Snake en Vanilla JS :

- HTML/CSS/JS sans framework.
- Sauvegarde du meilleur score en `localStorage` .

2. Calcul de Pi en Python :

- Méthode de Monte-Carlo.

3. Bot Discord Météo :

- Commande `/meteo <ville>` .

Risques & Considérations

- **Green IT** : Chaque prompt a un coût énergétique. Optimisez vos requêtes.
- **Sécurité** : Attention au *MCP poisoning* et aux vulnérabilités générées.
- **Éthique** : Les LLMs peuvent reproduire des biais (sexisme, racisme...). Soyez critiques.
- **Juridique** : Vérifiez les licences du code généré et respectez le RGPD.

Bilan & Perspectives

- Tour de table
- Le Vibe Coding est en constante évolution.
- Questions ?