

Nouveauté

Ref. IA-PYAUG

Développeur Python augmenté par l'IA

10 jours (70 heures)

Perfectionnement

Présentiel ou classe à distance

à partir de

6 000 € HT par personne (TVA en vigueur)

 Repas inclus en présentiel

Contenu mis à jour le 25/09/2026. Document téléchargé le 02/10/2026.

Objectifs

A l'issue de cette formation, vous serez capable de :

- Exploiter les assistants de code (Copilot, Cursor, Claude Code) pour accélérer le développement Python
- Générer, refactorer, tester, documenter et déboguer du code Python avec l'IA
- Sécuriser le code produit avec l'assistance de l'IA
- Intégrer des modèles de langage dans des applications Python (API, prompting, sorties structurées)
- Concevoir des applications LLM, des architectures RAG et des agents en Python
- Evaluer, sécuriser, déployer et optimiser des applications à base de LLM
- Inscrire votre pratique dans une démarche d'ingénierie IA conforme et durable.

Les compétences visées par cette formation font l'objet d'une évaluation.

À qui s'adresse cette formation ?

Développeurs Python expérimentés, ingénieurs logiciel, développeurs full-stack souhaitant intégrer l'IA à leurs applications, Lead Developers et référents techniques.

Prérequis

Disposer d'une expérience pratique du développement en Python (fonctions, classes, modules, gestion des dépendances) et être à l'aise avec un environnement de développement et la gestion de versions (Git). Aucune connaissance préalable en Intelligence Artificielle n'est requise. Avoir des notions d'API REST est un plus.

Programme

1 Pendant votre formation

Développeur Python augmenté par l'IA

Jour 1 – Matin

Situer l'IA Générative et l'agentique dans le métier de développeur

- Panorama IA, Machine Learning, IA générative et agents : ce que cela change pour le développeur
- Fonctionnement, capacités et limites des grands modèles de langage (LLM)
- Cartographie des usages de l'IA sur le cycle de développement
- Cadre éthique et juridique : AI Act, licences, confidentialité et propriété du code

Exemple de travaux pratiques (à titre indicatif)

- *Audit des opportunités d'usage de l'IA sur son propre flux de développement*

Jour 1 – Après-midi

Pratiquer le prompt engineering appliqué au code

- Principes du prompting : rôle, contexte, contraintes, exemples (few-shot)
- Modèles de prompts pour le développeur : génération, explication, refactoring, débogage
- Panorama des assistants : ChatGPT, Claude, Microsoft Copilot, Gemini, Le Chat de Mistral
- Confidentialité du code et des données transmises aux modèles

Exemple de travaux pratiques (à titre indicatif)

- *Constitution d'une bibliothèque de prompts de développement réutilisables*

Jour 2 – Matin

Exploiter un assistant de code dans son environnement

- Mise en place de GitHub Copilot, Cursor et Claude Code dans l'IDE
- Complétion, génération multi-fichiers et édition pilotée par l'IA
- Bonnes pratiques de contexte (fichiers de règles, conventions de projet)
- Comparaison des assistants et choix selon le contexte

Exemple de travaux pratiques (à titre indicatif)

- Configuration et prise en main d'un assistant de code sur un projet Python

Jour 2 – Après-midi

Générer et comprendre du code Python avec l'IA

- Génération de fonctions, classes et modules Python à partir de spécifications
- Explication et rétro-documentation de code existant ou hérité
- Revue critique : fiabilité, maîtrise de la dette technique, hallucinations
- Intégration des sorties dans un dépôt versionné (Git)

Exemple de travaux pratiques (à titre indicatif)

- Génération puis fiabilisation d'un module Python à partir d'un besoin

Jour 3 – Matin

Refactorer et moderniser du code Python avec l'IA

- Refactoring assisté : lisibilité, typage, conformité PEP 8
- Modernisation et migration (versions de Python, bibliothèques obsolètes)
- Détection de code smells et propositions d'amélioration
- Limites et garde-fous lors des transformations automatiques

Exemple de travaux pratiques (à titre indicatif)

- Refactoring guidé d'un module legacy

Jour 3 – Après-midi

Tester son code Python avec l'assistance de l'IA

- Génération de tests unitaires avec pytest à partir du code
- Couverture, cas limites et tests paramétrés assistés
- Démarche TDD augmentée par l'IA
- Validation humaine et fiabilité des tests générés

Exemple de travaux pratiques (à titre indicatif)

- Génération d'une suite de tests pytest et mesure de la couverture

Jour 4 – Matin

Documenter et déboguer avec l'IA

- Docstrings, annotations de type et génération de documentation
- Analyse d'erreurs et de traces (stack traces) assistée
- Diagnostic et résolution de bugs guidés par l'IA
- Génération d'exemples d'usage et de README

Exemple de travaux pratiques (à titre indicatif)

- Débogage assisté et documentation d'un projet Python

Jour 4 – Après-midi

Sécuriser le code produit avec l'IA

- Risques du code généré : vulnérabilités, dépendances, secrets
- Revue de sécurité assistée et référentiel OWASP
- Analyse de dépendances et de la chaîne d'approvisionnement logicielle
- Bonnes pratiques d'usage responsable des assistants de code

Exemple de travaux pratiques (à titre indicatif)

- *Audit de sécurité d'un code généré par l'IA*

Jour 5 – Matin

Appeler des modèles de langage depuis Python

- APIs et SDK des principaux fournisseurs (OpenAI, Anthropic, Mistral)
- Paramètres d'appel : température, jetons, streaming, gestion d'erreurs
- Maîtrise des coûts et des quotas
- Modèles ouverts et exécution locale (Ollama, Hugging Face)

Exemple de travaux pratiques (à titre indicatif)

- *Intégration d'un premier appel LLM dans un script Python*

Jour 5 – Après-midi

Industrialiser le prompting dans le code

- Gabarits de prompts et gestion des variables
- Sorties structurées (JSON) et validation des réponses
- Appel d'outils (function calling) depuis le code
- Stratégies de fiabilisation des réponses

Exemple de travaux pratiques (à titre indicatif)

- *Production de sorties structurées fiables à partir d'un LLM*

Jour 6 – Matin

Concevoir des applications LLM avec un framework

- Panorama des frameworks : LangChain, LlamaIndex
- Chaînes, mémoire et gestion du contexte
- Connexion à des outils et des sources de données
- Structuration et maintenabilité d'une application LLM

Exemple de travaux pratiques (à titre indicatif)

- *Construction d'une chaîne LLM outillée en Python*

Jour 6 – Après-midi

Mettre en oeuvre une architecture RAG en Python

- Principes du RAG : découpage, embeddings, bases vectorielles
- Recherche sémantique et reclassement des résultats
- Construction d'un pipeline RAG complet
- Evaluation de la pertinence et de la qualité des réponses

Exemple de travaux pratiques (à titre indicatif)

- Développement d'un assistant documentaire basé sur le RAG

Jour 7 – Matin

Développer un agent en Python

- Architecture d'un agent : raisonnement, mémoire, action
- Patterns agentiques : ReAct, planification, réflexion
- Usage d'outils, function calling et Model Context Protocol (MCP)
- Observabilité et traçage des décisions de l'agent

Exemple de travaux pratiques (à titre indicatif)

- Développement d'un agent outillé en Python

Jour 7 – Après-midi

Orchestrer des systèmes multi-agents

- Coordination et répartition des tâches entre agents
- Frameworks d'orchestration (LangGraph et équivalents)
- Gestion d'état, des erreurs et des défaillances en cascade
- Limites, coûts et fiabilité des architectures multi-agents

Exemple de travaux pratiques (à titre indicatif)

- Conception d'un workflow multi-agents simple

Jour 8 – Matin

Evaluer la qualité des applications à base de LLM

- Spécificités : non-déterminisme, hallucinations, dérive
- Evaluation par modèle juge (LLM-as-judge) et jeux d'évaluation
- Outils d'évaluation : RAGAS, DeepEval, promptfoo
- Tests de non-régression des applications LLM

Exemple de travaux pratiques (à titre indicatif)

- Mise en place d'un jeu d'évaluation reproductible

Jour 8 – Après-midi

Sécuriser les applications à base de LLM

- Menaces : injection de prompt, fuite de données, OWASP Top 10 LLM
- Garde-fous (guardrails), modération et filtrage

- Gestion des secrets, des permissions et des appels d'outils
- Documentation de la robustesse et conformité

Exemple de travaux pratiques (à titre indicatif)

- Renforcement de la sécurité d'une application LLM

Jour 9 – Matin

Déployer une application IA en Python

- Exposition d'une application via une API (FastAPI)
- Conteneurisation et préparation au déploiement
- Observabilité et traçage (LangSmith, journalisation)
- Bonnes pratiques LLMOps

Exemple de travaux pratiques (à titre indicatif)

- Exposition et instrumentation d'une application LLM

Jour 9 – Après-midi

Optimiser performances et coûts

- Maîtrise de la latence et des coûts d'inférence
- Mise en cache et optimisation des appels
- Choix d'architecture : RAG, fine-tuning, modèles ouverts
- Critères de sélection d'un modèle selon le besoin

Exemple de travaux pratiques (à titre indicatif)

- Optimisation du coût et de la latence d'une application LLM

Jour 10 – Matin

Réaliser un projet fil rouge (capstone)

- Mise en situation intégrée : développer augmenté par l'IA et intégrer l'IA dans l'application
- Travail sur un cas réaliste de bout en bout
- Restitution et revue par les pairs
- Consolidation des acquis des blocs précédents

Exemple de travaux pratiques (à titre indicatif)

- Projet intégrateur : une application Python augmentée par l'IA

Jour 10 – Après-midi

Ancrer une pratique de développement IA durable

- Bonnes pratiques d'ingénierie et de gouvernance (AI Act, ISO / IEC 42001)
- Intégration de l'IA dans les workflows d'équipe et la CI / CD
- Organisation de la veille et montée en compétence continue
- Bilan, évaluation des acquis et plan d'action personnel

Exemple de travaux pratiques (à titre indicatif)

- Construction d'une feuille de route personnelle de développeur augmenté

Financement

Plusieurs dispositifs de financement sont accessibles via les OPCO (Opérateurs de Compétences), organismes agréés par le ministère du Travail dont le rôle est d'accompagner, collecter et gérer les contributions des entreprises au titre du financement de la formation professionnelle.

Pour plus d'information, une équipe de gestionnaires SKOLAE Formation spécialisée vous accompagne dans le choix de vos formations et la gestion administrative.

[Nous contacter pour en savoir plus →](#)

Modalités pédagogiques

Formation délivrée en présentiel ou distanciel* (blended-learning, e-learning, classe virtuelle, présentiel à distance).

Le formateur alterne entre méthode** démonstrative, interrogative et active (via des travaux pratiques et/ou des mises en situation).

Variables suivant les formations, les moyens pédagogiques mis en oeuvre sont :

- Ordinateurs Mac ou PC (sauf pour certains cours de l'offre Management), connexion internet fibre, tableau blanc ou paperboard, vidéoprojecteur ou écran tactile interactif (pour le distanciel)
- Environnements de formation installés sur les postes de travail ou en ligne
- Supports de cours et exercices

En cas de formation intra sur site externe à M2i, le client s'assure et s'engage également à avoir toutes les ressources matérielles pédagogiques nécessaires (équipements informatiques...) au bon déroulement de l'action de formation visée conformément aux prérequis indiqués dans le programme de formation communiqué.

* nous consulter pour la faisabilité en distanciel

** ratio variable selon le cours suivi

[Accessibilité aux personnes en situation de handicap](#)

Détails de la session

Du 22 février 2027 au 5 mars 2027 (10 jours / 70 heures)

Paris La Défense

Du 22 février 2027 au 5 mars 2027

Paris La Défense

- 22 février 2027 9h00 - 17h00
- 23 février 2027 9h00 - 17h00
- 24 février 2027 9h00 - 17h00
- 25 février 2027 9h00 - 17h00
- 26 février 2027 9h00 - 17h00
- 27 février 2027 9h00 - 17h00
- 28 février 2027 9h00 - 17h00
- 1 mars 2027 9h00 - 17h00
- 2 mars 2027 9h00 - 17h00
- 3 mars 2027 9h00 - 17h00
- 4 mars 2027 9h00 - 17h00
- 5 mars 2027 9h00 - 17h00