+33 (0)1 48 10 09 46
formation@ambient-it.net
140 Avenue Jean Lolive 93500 Pantin - France

Mis à jour le 06/06/2025

S'inscrire

Formation Python Perfectionnement

4 jours (28 heures)

PRÉSENTATION

Notre formation Python perfectionnement vous apprendra à développer vos compétences dans le langage de programmation pour développer des applications plus performantes et optimisées à la pointe des évolutions du langage.

Cette formation avancée sur Python s'adresse à tout développeur maitrisant déjà les concepts de base de Python. Cette formation vous enseignera les concepts les plus avancés du langage, comme le typage évolué, la métaprogrammation, pattern matching.

Vous explorerez les dernières innovations de Python (annotations paresseuses, métaclasses avancées, Task Groups async, Template Strings), et consoliderez votre savoir-faire en programmation fonctionnelle et orientée objet moderne grâce aux Data Classes et Protocols typés.

Vous apprendrez à optimiser radicalement vos applications : parallélisme performant avec multiprocessing et Threads (GIL 2.0), programmation asynchrone avancée, ainsi que profiling et monitoring temps réel pour garantir performance et fiabilité de vos programmes en production.

Côté industrialisation, vous maîtriserez les meilleures pratiques pour packager et déployer vos projets Python grâce à Poetry, PyInstaller, Docker et Kubernetes, tout en sécurisant votre distribution via Sigstore et dépôts internes (PyPI privé, Artifactory).

Enfin, vous exploiterez pleinement l'écosystème Python (Polars, AutoML, FastAPI, Scapy NG, cryptographie) pour livrer des applications robustes en data science, machine learning, cybersécurité et web.

Comme tous nos programmes, les exercices pratiques sont au cœur de cette formation, ce qui vous donnera les compétences opérationnelles nécessaires, dans la dernière version du langage (Python 3.13 à la date de cet article).

OBJECTIFS

- Approfondir la connaissance des concepts avancés de Python
- Utiliser les techniques avancées du langage Python
- Optimiser les performances de vos programmes à l'aide du monitoring et du parallélisme
- Packager et déployer ses artefacts Python
- Exploiter des librairies contribuant au succès du langage (data science & machine Learning, cybersécurité, développement web, logiciels et outils...)

PUBLIC VISÉ

- Ingénieurs et développeurs

Pré-requis

- Disposer de bonnes connaissances en développement Python

Programme de notre Formation Python Perfectionnement

Nouveautés majeures de Python (2021–2025)

- Python 3.10 (2021)
  - Pattern Matching (PEP 634) : introduction du match-case
  - Gestion d’erreurs améliorée avec des messages d’erreurs plus informatifs
  - Annotation de types avancées (| pour types alternatifs)
- Python 3.11 (2022)
  - Gestion améliorée des exceptions et traceback détaillé (PEP 657)
  - Typing Self (PEP 673) : annotations simplifiées dans les méthodes de classe
- Python 3.12 (2023)
  - Typage amélioré : annotations strictes avec des améliorations sur les génériques
- Python 3.13 (2024)
  - Task Groups (asyncio, PEP 671) pour gestion améliorée de tâches asynchrones
  - Paramètres par défaut évalués tardivement (PEP 649) simplifiant les annotations avancées
  - Métaprogrammation renforcée : classes dynamiques avancées (PEP 696)
- Le futur Python 3.14 (2025)
  - Template strings intégrées (t"") – simplifie la génération dynamique de chaînes.
  - Annotations évaluées paresseusement – meilleure gestion des références circulaires.
  - Débogage externe sécurisé – attachement facilité d’un débogueur externe.
  - Module compression.zstd intégré – compression rapide intégrée à la bibliothèque standard.
  - UUID versions 6 à 8 supportées – génération plus rapide et sécurisée.
  - Passage à Sigstore – signature sécurisée des paquets, remplaçant les signatures PGP obsolètes.

Python 3.13 et idiomes avancés

- Pattern matching avancé (PEP 634)
- Annotations strictes (PEP 695)
- Gestion avancée des erreurs (PEP 654)
- Atelier pratique : Structure initiale d’une plateforme de monitoring intelligent.

Syntaxe moderne et typage évolué

- PEP 695 : déclarer des génériques allégés et explicites)
- Dataclasses, slots=True et frozen=True pour des objets légers
- Structural Pattern Matching pour des flux de contrôle expressifs
- Bonnes pratiques de type-hints et vérification statique (mypy, pyright)

Programmation fonctionnelle sophistiquée

- Fonctions d’ordre supérieur
- Closures optimisées
- Décorateurs génériques (PEP 681)
- Atelier pratique : Implémenter des décorateurs dynamiques pour tracer l’application.

Python orienté objet : Data classes et Protocols

- Data Classes améliorées (PEP 681)
- Protocols et interfaces typées (PEP 544)
- Héritage multiple moderne et mixins
- Atelier pratique : Développer la couche métier du monitoring avec data classes.

Métaprogrammation & conception orientée objet poussée

- Décorateurs imbriqués, classes décoratrices et descriptors
- Context managers avancés
- Métaclasses : génération de classes et hooks
- API Attrs/Pydantic v2 pour déclarer des modèles performants

Métaclasses et plugins dynamiques

- Principe avancé des métaclasses
- Création dynamique de classes (PEP 696)
- Plugins automatisés
- Atelier pratique : Architecture extensible via métaclasses pour plugins.

Context Managers avancés et automatisation

- Context Managers personnalisés optimisés
- Async Context Managers

- Gestion des ressources critiques (asyncio amélioré)
- Atelier pratique : Context manager intelligent pour accès base de données.

Concurrence avancée : Threads et GIL 2.0

- GIL optimisé (nouvelle implémentation CPython)
- ThreadPoolExecutor haute performance
- Gestion avancée concurrence
- Atelier pratique : Traitement concurrent de flux en temps réel de données.

Parallélisme et multiprocessing haute performance

- Multiprocessing Pool optimisé (Process Pools)
- Partage mémoire inter-process avancé
- Celery nouvelle génération, Dask distribué
- Atelier pratique : Traitement distribué intensif des données sous monitoring.

Programmation asynchrone innovante avec asyncio

- Async/Await avancé (Python 3.13)
- Nouveautés asyncio (PEP 671 - Task Groups)
- Combinaison d’async et multi-processus
- Atelier pratique : Agrégation non-bloquante des données externes via API.

Profiling et optimisation continue du code

- Profiling temps réel (Py-Spy, Scalene)
- Optimisation mémoire avec tracemalloc amélioré
- Techniques modernes : memoization avancée, compilation JIT (Numba)
- Atelier pratique : Optimiser le cœur algorithmique de la plateforme.

Structuration et gestion de projet

- Organisation modulaire : dossiers et fichiers (setup.py, pyproject.toml)
- Gestion moderne des dépendances : Poetry, Pipenv, gestion explicite des versions
- Virtualisation d’environnements : venv, Conda pour isoler les dépendances
- Atelier pratique : Initialiser et structurer proprement un projet Python avec Poetry et un environnement virtuel dédié.

Packaging avancé

- Poetry & Pyproject.toml : packaging simplifié, fiable et reproductible
- Création de Wheel : distribution optimisée (.whl vs .tar.gz)
- Construction d’exécutables : PyInstaller, cx_Freeze pour exécutable autonome
- Atelier pratique : Créer un artefact distribuable (Wheel) et un exécutable autonome avec PyInstaller.

Monitoring et analyse de performance

- Profiler intégré
- Sampling profilers (py-spy, Scalene) pour la production
- Traque mémoire
- Visualisation des traces (snakeviz, speedscope)

Déploiement et Distribution sécurisée

- Publication publique sur PyPI : distribution ouverte (twine, CI/CD)
- Index privés : Artifactory, Nexus, PyPI privé interne (sécurisé avec authentification)
- Signature et sécurité : Sigstore, vérification des artefacts avec clés sécurisées
- Atelier pratique : Publier et sécuriser un paquet Python sur un dépôt interne avec signature cryptographique (Sigstore).

Data science : Analyse avancée & visualisation interactive

- Pandas 2.x optimisé
- Polars (dataframes haute performance)
- Visualisations interactives (Plotly Dash, Streamlit avancé)
- Atelier pratique : Tableau de bord interactif des indicateurs clés.

Données volumineuses à vitesse éclair

- Benchmarks Polars vs Pandas : gains ×5 à ×20 sur les agrégations
- APIs Lazy Polars et query optimisation
- Gestion efficace des formats colonne (Parquet, Arrow), impact sur l’IO
- Atelier pratique : migrer un pipeline Pandas vers Polars et mesurer le speed-up

Machine Learning de dernière génération

- Nouveautés scikit-learn 1.5 (hist-GBDT, accélération CPU/GPU)
- Intégration des modèles vers ONNX / Pydantic pour la validation
- Aperçu des frameworks ML avancés (PyTorch 2.x, Lightning)

Data science : Analyse avancée & visualisation interactive

- Introduction au Machine Learning

- AutoML (PyCaret, Auto-sklearn 2.0)
- Intégration du Deep Learning (TensorFlow 3.x, PyTorch 3.0)
- Modèles prédictifs intelligents, explicabilité (SHAP)
- Atelier pratique : Détection intelligente d’anomalies avec AutoM

Développement Web : APIs performantes et scalables

- FastAPI avancé (async, websockets)
- Sécurité avancée (OAuth2, JWT nouvelle génération)
- Microservices, API Gateway (Typer, FastAPI Gateway)
- Atelier pratique : API scalable et sécurisée pour exposer les résultats.

Cybersécurité et bibliothèques innovantes

- Sécurité applicative moderne (OWASP 2025, cryptography 3.0)
- Python et cybersécurité avancée (Scapy NG, Requests sécurisées, paramétrisation sécurisée)
- Veille technologique permanente (PyCon 2025, GitHub Trends)
- Atelier pratique : Sécurisation complète et veille active sur la plateforme.

Sociétés concernées

Cette formation s’adresse à la fois aux particuliers ainsi qu’aux entreprises, petites ou grandes, souhaitant former ses équipes à une nouvelle technologie informatique avancée ou bien à acquérir des connaissances métiers spécifiques ou des méthodes modernes.

Positionnement à l'entrée en formation

Le positionnement à l'entrée en formation respecte les critères qualité Qualiopi. Dès son inscription définitive, l'apprenant reçoit un questionnaire d’auto-évaluation nous permettant d’apprécier son niveau estimé sur différents types de technologies, ses attentes et objectifs personnels quant à la formation à venir, dans les limites imposées par le format sélectionné. Ce questionnaire nous permet également d’anticiper certaines difficultés de connexion ou de sécurité interne en entreprise (intraentreprise ou classe virtuelle) qui pourraient être problématiques pour le suivi et le bon déroulement de la session de formation.

Méthodes pédagogiques

Stage Pratique : 60% Pratique, 40% Théorie. Support de la formation distribué au format numérique à tous les participants.

Organisation

Le cours alterne les apports théoriques du formateur soutenus par des exemples et des séances de réflexions, et de travail en groupe.

Validation

À la fin de la session, un questionnaire à choix multiples permet de vérifier l’acquisition correcte

des compétences.

Sanction

Une attestation sera remise à chaque stagiaire qui aura suivi la totalité de la formation.