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

Updated 06/06/2025

Sign up

Python Perfectionnement training

4 days (28 hours)

PRESENTATION

Our advanced Python training course will teach you to develop your skills in the programming language so you can develop more powerful, optimized applications at the cutting edge of language developments.

This advanced Python training course is designed for developers who have already mastered the basic concepts of Python. This course will teach you the most advanced concepts of the language, such as advanced typing, metaprogramming and pattern matching.

You'll explore the latest Python innovations (lazy annotations, advanced metaclasses, async Task Groups, Template Strings), and consolidate your know-how in modern functional and object-oriented programming with typed Data Classes and Protocols.

You'll learn how to radically optimize your applications: high-performance parallelism with multiprocessing and Threads (GIL 2.0), advanced asynchronous programming, as well as real-time profiling and monitoring to guarantee the performance and reliability of your programs in production.

On the industrialization side, you'll master best practices for packaging and deploying your Python projects using Poetry, PyInstaller, Docker and Kubernetes, while securing your distribution via Sigstore and internal repositories (private PyPI, Artifactory).

Finally, you'll make full use of the Python ecosystem (Polars, AutoML, FastAPI, Scapy NG, cryptography) to deliver robust applications in data science, machine learning, cybersecurity and the web.

As with all our programs, practical exercises are at the heart of this training, giving you the necessary operational skills in the latest version of the language (Python 3.13 at the time of writing).

OBJECTIVES

- Deepen your knowledge of advanced Python concepts
- Use advanced Python language techniques
- Optimize program performance through monitoring and parallelism
- Packaging and deploying Python artifacts
- Use libraries that contribute to the success of the language (data science & machine learning, cybersecurity, web development, software and tools, etc.).

TARGET AUDIENCE

- Engineers and developers

Prerequisites

- Good knowledge of Python development

Program of our Python Perfectionnement training course

Major new features in Python (2021-2025)

- Python 3.10 (2021)
  - Pattern Matching (PEP 634): introduction of the match-case
  - Improved error handling with more informative error messages
  - Advanced type annotation (| for alternative types)
- Python 3.11 (2022)
  - Enhanced exception handling and detailed traceback (PEP 657)
  - Typing Self (PEP 673): simplified annotations in class methods
- Python 3.12 (2023)
  - Enhanced typing: strict annotations with improved generics
- Python 3.13 (2024)
  - Task Groups (asyncio, PEP 671) for improved management of asynchronous tasks
  - Late-evaluated default settings (PEP 649) simplify advanced annotations
  - Enhanced metaprogramming: advanced dynamic classes (PEP 696)
- The future of Python 3.14 (2025)
  - Template strings integrated (t"") - simplifies dynamic string generation.
  - Annotations lazily evaluated - better management of circular references.
  - Secure external debugging - easy attachment of an external debugger.
  - Integrated .zstd compression module - fast compression integrated into the standard library.
  - UUID versions 6 to 8 supported - faster, more secure generation.
  - Switch to Sigstore - secure packet signing, replacing obsolete PGP signatures.

Python 3.13 and advanced idioms

- Advanced pattern matching (PEP 634)
- Strict annotations (PEP 695)
- Advanced error handling (PEP 654)
- Practical workshop: Initial structure of an intelligent monitoring platform.

Modern syntax and advanced typing

- PEP 695: declare lean and explicit generics)
- Dataclasses, slots=True and frozen=True for lightweight objects
- Structural Pattern Matching for expressive control flows
- Best practices for type-hints and static verification (mypy, pyright)

Sophisticated functional programming

- Higher-order functions
- Optimized closures
- Generic decorators (PEP 681)
- Practical workshop: Implementing dynamic decorators for application layout.

Object-oriented Python: Data classes and Protocols

- Enhanced Data Classes (PEP 681)
- Protocols and typed interfaces (PEP 544)
- Modern multiple heritage and mixins
- Practical workshop: Developing the monitoring business layer with data classes.

Metaprogramming & advanced object-oriented design

- Nested decorators, decorator classes and descriptors
- Advanced context managers
- Metaclasses: class generation and hooks
- Attrs/Pydantic v2 API for declaring high-performance models

Metaclasses and dynamic plugins

- Advanced metaclass principle
- Dynamic class creation (PEP 696)
- Automated plugins
- Practical workshop: Extensible architecture via plug-in metaclasses.

Advanced Context Managers and automation

- Customized, optimized Context Managers
- Async Context Managers

- Critical resource management (enhanced asyncio)
- Practical workshop: Intelligent context manager for database access.

Advanced competition: Threads and GIL 2.0

- Optimized GIL (new CPython implementation)
- High-performance ThreadPoolExecutor
- Advanced competition management
- Practical workshop: Concurrent processing of real-time data streams.

Parallelism and high-performance multiprocessing

- Optimized Multiprocessing Pools (Process Pools)
- Advanced inter-process memory sharing
- Celery new generation, Dask distributed
- Practical workshop: Intensive distributed data processing with monitoring.

Innovative asynchronous programming with asyncio

- Advanced async/await (Python 3.13)
- Asyncio news (PEP 671 - Task Groups)
- Combining async and multi-processing
- Practical workshop: Non-blocking aggregation of external data via API.

Profiling and continuous code optimization

- Real-time profiling (Py-Spy, Scalene)
- Memory optimization with enhanced tracemalloc
- Modern techniques: advanced memoization, JIT compilation (Numba)
- Practical workshop: Optimizing the platform's algorithmic core.

Project structuring and management

- Modular organization: folders and files (setup.py, pyproject.toml)
- Modern dependency management: Poetry, Pipenv, explicit version management
- Virtualizing environments: venv, Conda for isolating dependencies
- Practical workshop: Initializing and structuring a Python project with Poetry and a dedicated virtual environment.

Advanced packaging

- Poetry & Pyproject.toml: simplified, reliable and reproducible packaging
- Wheel creation: optimized distribution (.whl vs .tar.gz)
- Building executables: PyInstaller, cx_Freeze for standalone executables
- Practical workshop: Creating a distributable artifact (Wheel) and a standalone executable with PyInstaller.

Performance monitoring and analysis

- Integrated profiler
- Sampling profilers (py-spy, Scalene) for production
- Memory tracking
- Track visualization (snakeviz, speedscope)

Deployment and secure distribution

- Public publication on PyPI: open distribution (twine, CI/CD)
- Private indexes: Artifactory, Nexus, internal private PyPI (secured with authentication)
- Signature and security: Sigstore, artifact verification with secure keys
- Practical workshop: publishing and securing a Python package on an internal repository with cryptographic signature (Sigstore).

Data science: advanced analysis & interactive visualization

- Pandas 2.x optimized
- Thrillers (high-performance dataframes)
- Interactive visualizations (Plotly Dash, advanced Streamlit)
- Practical workshop: Interactive dashboard of key indicators.

High-volume data at lightning speed

- Polars vs Pandas benchmarks: ×5 to ×20 gains on aggregations
- Lazy Polars APIs and query optimization
- Efficient management of column formats (Parquet, Arrow), impact on IO
- Practical workshop: migrating a Pandas pipeline to Polars and measuring speed-up

Latest-generation Machine Learning

- What's new in scikit-learn 1.5 (hist-GBDT, CPU/GPU acceleration)
- Integration of models into ONNX / Pydantic for validation
- Overview of advanced ML frameworks (PyTorch 2.x, Lightning)

Data science: advanced analysis & interactive visualization

- Introduction to Machine Learning

- AutoML (PyCaret, Auto-sklearn 2.0)
- Deep Learning integration (TensorFlow 3.x, PyTorch 3.0)
- Intelligent predictive models, explainability (SHAP)
- Workshop: Intelligent fault detection with AutoM

Web development: high-performance, scalable APIs

- Advanced FastAPI (async, websockets)
- Advanced security (OAuth2, next-generation JWT)
- Microservices, API Gateway (Typer, FastAPI Gateway)
- Practical workshop: scalable, secure API for displaying results.

Cybersecurity and innovative libraries

- Modern application security (OWASP 2025, cryptography 3.0)
- Python and advanced cybersecurity (Scapy NG, secure requests, secure parameterization)
- Constant technology watch (PyCon 2025, GitHub Trends)
- Practical workshop: Complete security and active monitoring of the platform.

Companies concerned

This course is aimed at both individuals and companies, large or small, wishing to train their teams in a new advanced computer technology, or to acquire specific business knowledge or modern methods.

Positioning on entry to training

Positioning at the start of training complies with Qualiopi quality criteria. As soon as registration is finalized, the learner receives a self-assessment questionnaire enabling us to assess his or her estimated level of proficiency in different types of technology, as well as his or her expectations and personal objectives with regard to the training to come, within the limits imposed by the selected format. This questionnaire also enables us to anticipate any connection or security difficulties within the company (intra-company or virtual classroom) which could be problematic for the follow-up and smooth running of the training session.

Teaching methods

Practical course: 60% Practical, 40% Theory. Training material distributed in digital format to all participants.

Organization

The course alternates theoretical input from the trainer, supported by examples, with brainstorming sessions and group work.

Validation

At the end of the session, a multiple-choice questionnaire is used to check correct acquisition.

skills.

Sanction

A certificate will be issued to each trainee who completes the course.