+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 : Programmation Objet

5 jours (35 heures)

Présentation

La formation Python, Programmation Orientée Objet, conçue pour développer rapidement vos compétences en Python. Structurée sur cinq jours, elle combine théorie, pratique et un projet fil rouge concret, permettant une immersion complète dans les fondamentaux du langage Python.

Vous maîtriserez d'abord les bases du langage : syntaxe claire, gestion efficace des variables et structures de contrôle essentielles. Dès le second jour, vous aborderez les fonctions, les modules et la gestion des fichiers pour structurer efficacement votre code.

Le cœur de la formation se concentre ensuite sur la programmation orientée objet. Vous apprendrez à concevoir des classes robustes, à gérer l'héritage, et à utiliser les méthodes spéciales et décorateurs qui rendent votre code élégant, lisible et réutilisable.

Pour renforcer vos compétences pratiques, vous développerez une interface graphique intuitive avec Tkinter, connectée directement à vos classes Python. Ce projet évolutif vous permettra de visualiser concrètement les résultats de votre apprentissage.

Enfin, vous acquerrez les meilleures pratiques professionnelles grâce aux outils de tests unitaires et d'analyse de code, garantissant un code Python de haute qualité, prêt pour une utilisation en contexte professionnel.

À l'issue de la formation, le stagiaire sera capable d'utiliser les principales fonctionnalités du langage de programmation Python pour développer des applications multiplateformes.

Dans cette formation, comme dans toutes nos formations que nous vous proposons, nous utiliserons la dernière version stable (Python 3.13 à la date de cet article).

Objectifs

- Maîtriser la syntaxe du langage Python
- Acquérir les notions essentielles de la Programmation Objet
- Concevoir des interfaces graphiques
- Savoir appliquer les fonctionnalités des modules Python
- Maîtriser les outils de test et d'évaluation de programme

Public visé

- Développeurs
- Architectes
- Ingénieurs
- Chef de projets en informatique

Pré-requis

- Connaissance de base en programmation.

Pré-requis

- L’interpréteur Python installé
- Un environnement de développement comme PyCharm ou Visual Studio Code
- Un lecteur PDF et un logiciel d'archivage (.zip)
- Les droits d'accès au réseau pour installer des bibliothèques

Programme de notre formation Python : Programmation Objet

Introduction à Python

- Installation de Python et des outils de développement
- Syntaxe de base et conventions (PEP8)
- Modes d'exécution : script et interactif
- Utilisation de l'interpréteur et des IDE
- Python et l’approche fonctionnelle
- Atelier pratique : Installer l'environnement Python et exécuter un script simple.

Variables et types de données

- Types de données primitifs : int, float, str, bool
- Chaînes de caractères (str) immuables
- La gestion des dates
- Conversions de types et opérations de base
- Entrées utilisateur avec input()

- Utilisation de f-strings pour le formatage
- Typage dynamique : Duck Typing
- Atelier pratique : Créer un script demandant le nom et la date de naissance, calculer l'âge de l'utilisateur, puis afficher un message personnalisé.

Structures de contrôle

- Instructions conditionnelles if, elif, else
- Boucles for et while
- Utilisation de break et continue
- Compréhensions de listes
- Atelier pratique : Écrire un programme qui affiche les nombres premiers en utilisant une boucle.

Fonctions en Python

- Définition et appel de fonctions
- Paramètres et valeurs de retour
- Portée des variables (locale vs globale)
- Fonctions lambda et expressions anonymes
- Atelier pratique : Créer une fonction qui calcule la factorielle d'un nombre donné.

Concevoir du code Python compatible multiplateforme

- Utiliser des bibliothèques Python standards pour garantir la portabilité
- Éviter les chemins de fichiers codés en dur, préférer os.path ou pathlib
- Gérer les dépendances avec des environnements virtuels (venv, pipenv)
- Tester régulièrement son application sur différents systèmes (Windows, macOS, Linux)
- Respecter les bonnes pratiques de structuration et de séparation du code (logique métier / interface)
- Atelier pratique : Adapter et tester le projet Python fil rouge sur deux plateformes différentes afin d’identifier et corriger d’éventuelles incompatibilités.

Introduction aux expressions régulières

- Présentation du concept des regex
- Recherche, extraction et remplacement de motifs textuels
- Build, test, et debug facilement : regex101 & regexr.com
- Performances, optimisation des expressions régulières
- Atelier pratique : Utiliser une regex pour extraire automatiquement des adresses e-mail et numéros de téléphone à partir d’un fichier texte, puis valider et optimiser l’expression régulière.

Modules et packages

- Structurer le code (modules, packages, imports)
- Importation de modules standards et tiers
- Création et utilisation de modules personnalisés

- Gestion des dépendances avec pip
- Utilisation d'environnements virtuels
- Atelier pratique : Organiser le code en modules séparés pour améliorer la lisibilité et la réutilisabilité.

Concepts de la POO

- Classes et objets
- Attributs et méthodes
- Encapsulation et abstraction
- Visibilité
- Avantages de la POO
- Atelier pratique : Définir une classe Personne avec des attributs nom et âge, et une méthode pour afficher ces informations.

Héritage et polymorphisme

- Héritage simple et multiple
- Redéfinition de méthodes (overriding)
- Utilisation de super()
- Polymorphisme, et interfaces
- Atelier pratique : Créer une classe Employé héritant de Personne, ajoutant un attribut poste et redéfinissant la méthode d'affichage.

Méthodes spéciales et décorateurs

- Méthodes magiques __init__, __str__, __repr__
- Décorateurs @property, @classmethod, @staticmethod
- Utilisation de __slots__ pour optimiser la mémoire
- Introduction aux dataclasses
- Atelier pratique : Implémenter une classe Produit avec des méthodes spéciales pour afficher et comparer les objets.

Les collections intégrées en Python

- Listes (list)
- Tuples (tuple)
- Dictionnaires (dict)
- Ensembles (set)
- Frozen sets (frozenset)
- Modules spécialisés :
  - deque : files double-ended (queues)
  - namedtuple : tuples nommés pour une meilleure lisibilité
  - OrderedDict : dictionnaire garantissant explicitement l'ordre
  - Counter : compte efficacement les occurrences d'éléments

Sérialisation et désérialisation de données en Python

- Comprendre les principes de sérialisation (conversion objet ? flux de données)
- Utilisation de formats courants : JSON (json), Pickle (pickle)
- Choisir le format adapté : sécurité, lisibilité, performance
- Sérialisation de structures de données complexes (listes, dictionnaires, objets)
- Bonnes pratiques : gestion des erreurs, compatibilité multiplateforme
- Atelier pratique : Sérialiser et désérialiser une liste d'objets « Contact » en JSON pour sauvegarder et recharger simplement les données de l'application fil rouge.

Gestion des fichiers et des exceptions

- Lecture et écriture de fichiers texte
- Utilisation du gestionnaire de contexte with
- Bonnes pratiques pour intercepter et gérer des exceptions
- Gestion des exceptions avec try, except, finally
- Quand utiliser les exceptions ?
- Création d'exceptions personnalisées
- Logger et tracer les erreurs efficacement
- Atelier pratique : Écrire un programme qui lit un fichier texte et compte le nombre de mots, en gérant les erreurs potentielles.

Accès aux bases de données avec Python

- Présentation des bibliothèques Python courantes : SQLite (sqlite3), SQLAlchemy
- Connexion à une base de données et exécution de requêtes SQL simples
- CRUD : Création, lecture, mise à jour, suppression de données
- Gestion sécurisée des transactions (commit, rollback, exceptions)
- Bonnes pratiques : prévention des injections SQL, gestion des connexions
- Atelier pratique : Intégrer une base SQLite au projet fil rouge pour stocker, récupérer et gérer efficacement les contacts.

Introduction à Tkinter

- Présentation des bibliothèques graphiques Python
  - Tkinter
  - PyQt
  - Kivy
- Création d'une fenêtre principale
- Ajout de widgets de base : Label, Button, Entry
- Organisation des widgets avec pack, grid
- Gestion des événements simples
- Atelier pratique : Créer une interface simple avec un champ de saisie et un bouton affichant le texte saisi.

Construction d'une interface utilisateur

- Utilisation de Frame pour structurer l'interface
- Ajout de menus et de barres d'outils

- Gestion des événements complexes
- Personnalisation de l'apparence des widgets
- Atelier pratique : Développer une interface pour un gestionnaire de tâches avec ajout, suppression et affichage des tâches.

Intégration de la logique métier

- Connexion de l'interface aux classes Python
- Mise à jour dynamique de l'interface
- Gestion des erreurs et des validations
- Séparation du code en modules MVC
- Atelier pratique : Intégrer la logique du gestionnaire de contacts à l'interface graphique, permettant l'ajout et la suppression de contacts.

Tests unitaires avec unittest

- Création de cas de test
- Assertions courantes
- Organisation des tests en suites
- Exécution automatique des tests
- Atelier pratique : Écrire des tests unitaires pour les méthodes du gestionnaire de contacts, vérifiant l'ajout et la suppression de contacts.

Analyse de code et bonnes pratiques

- Utilisation de pylint et flake8 pour l'analyse statique
- Mesure de la couverture de code avec coverage.py
- Profilage des performances avec cProfile
- Débogage avancé avec pdb et les IDE modernes
- Les modules de tests unitaires
- Documentation du code avec des docstrings
- Atelier pratique : Analyser le code du projet avec pylint, corriger les problèmes identifiés et générer un rapport de couverture de tests.

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.

Organisme de formation enregistré sous le numéro 11 75 54743 75. Cet enregistrement ne vaut pas agrément de l'État.
© Ambient IT 2015-2025. Tous droits réservés. Paris, France - Suisse - Belgique - Luxembourg