# Titre

Python et la programmation objet

# Contenu validé

Oui

# Catch phrase (2 lignes max)

Découvrez la programmation orientée objet en Python pour réaliser des programmes multiplateformes robustes et fiables.

# Description (10 lignes max)

Python est un langage de programmation multiplateforme permettant le développement d'une grande variété d'applications. Dans cette formation, vous verrez comment maîtriser suffisamment python pour réaliser des programmes plus “pythoniques”.

Après avoir (re)vu la syntaxe du langage et (re)découvert les mécanismes de la programmation objet, vous aurez une base de connaissance solide. La suite du cours se concentrera sur les bonnes pratiques (évaluation de la qualité d’un code, tests, partage d’environnement) et l’utilisation de différentes bibliothèques pour gérer les cas de la vie réelle (manipulation de bases de données, lecture / écriture de fichiers, gestion d’une arborescence de dossiers / fichiers)

# Identifiant technique

pyobjet

# Domaine

développement

# Sous-domaine

Python

# Url

/formations/developpement/python-objet

# Ordre dans la page

2

# Prochaines sessions

## Lyon

21-22 juin 2021

## Lille

14-15 juin 2021

## Paris

28-29 juin 2021

# Équilibre théorie / pratique

35% théorie / 65% pratique

# Public visé / participants

* Développeurs
* Ingénieurs
* Chercheurs
* chefs de projets proches du développement

# Prérequis

* Être à l’aise avec un langage de programmation proche (C / C++ / Java / PHP / …) et connaitre les concepts de variables, types, fonctions, ...
* La connaissance de Python n’est pas nécessaire, mais souhaitable. La connaissance préalable de la programmation orientée objet n’est pas nécessaire.

# Durée

5 jours, soit 35h

# Objectifs pédagogiques

* Maîtriser la syntaxe du langage Python
* Acquérir les grandes notions de la programmation objet
* Choisir et utiliser les modules Python pour résoudre un problème spécifique
* Concevoir des interfaces graphiques
* Analyser un projet Python pour déterminer sa qualité de code
* Utiliser des tests unitaires pour améliorer la qualité d’un code

# Tarifs

1²650 € HT (pauses et repas inclus)

# 

# Plan de formation / programme

## **Le langage Python**

### Présentation / historique et contexte / philosophie

### Cas d’utilisation et de non utilisation

### Présentation des outils de développement

TP : installation d’un interpréteur Python, d’un éditeur de texte et familiarisation avec les outils

## **Syntaxe basique de Python**

### Variables et types classiques (assignation, modification…)

* Affichage des variables
* Conventions et règles de nommage.

### La manipulation des différents types

* Numériques
* chaînes de caractères
* Autres conteneurs (listes, tuple et dictionnaires)

### Les blocs, les commentaires.

* Les conditions (if/elif/else) et la manipulation de variables booléennes (création, composition)
* Les boucles while et for (gestion de l’itération avec break / continue)
* La fonction range.

### La gestion d’erreur et la compréhension des exceptions

### Création, documentation et utilisation de fonctions.

* Portée des variables et la règle LEGB
* Les lambda expression
* Les générateurs / itérateurs

### La structuration du code en modules.

### L'utilisation des fichiers.

TP : exercices permettant de se familiariser avec le langage (syntaxe, itérations et conditions, ouverture de fichiers) et d’illustrer la spécificité d’un code Pythonique

## **Qualité de code (QA)**

### Les environnements virtuels pour gérer facilement les projets

* Le concept et les cas d’usages des environnements virtuels
* Installation de bibliothèques tierces avec pip
* Création et évolution d’environnements virtuels

### Les outils d’analyse statique et l’annotation de type

* Présentation du concept
* Présentation des outils (pylint, mypy)

### Le formatage

* Présentation de la PEP 8
* Utilisation d’autoformatter (black, autopep8)

### La gestion de la documentation

* Écriture de la documentation (docstring, formattage)
* Les outils d’extraction de documentation

### Les tests unitaires

* Le concept de tests unitaires
* Les outils de tests unitaires (unittest, pytest) en python, notion de couverture de tests et génération de rapports
* Présentation du concept de développement piloté par les tests (TDD)
* L'automatisation des tests, l'agrégation de tests.
* Intégration dans un pipeline d’intégration continue (CI)

### Débogage

* Le concept de déboguer du code
* Le débugueur textuel de python
* Les débugueurs graphiques

TP : Configurer son éditeur de code pour intégrer des outils de QA, ajout de tests unitaires sur des codes précédents

## **Présentation de la Programmation Orientée Objet (POO)**

### Les principes du paradigme Objet.

### La définition d'un objet (état, comportement, identité).

### La notion de classe, d'attributs et de méthodes.

### La communication entre les objets.

* L'héritage et la notion de polymorphisme.
* Composition de classes.

### Modélisation de programmes avec l’UML

* Les diagrammes de classes, de séquences, d'activités…
* Notion de modèle de conception (Design Pattern).

**TP :** *Modéliser un cas concret en UML (identification des classes et de leurs relations)*

## **Programmation Objet en Python**

### Les particularités du modèle Objet de Python.

### L'écriture de classes et leur instanciation.

* Les méthodes et attributs en pratique
* La modification d’attribut
* Le paramètre “self”
* Les méthodes magiques pour modifier le comportement de ses classes

### L'héritage et le polymorphisme.

### Le duck typing

### Création d’exceptions spécifiques

**TP :** *Pratique des différents concepts à l’aide d’un ensemble d’exercices simples.*

## **Les bibliothèques**

### Qu’est-ce qu’une bibliothèque

### La bibliothèque standard et ses différents modules

* Manipulation de fichiers
* Manipulation réseau
* Expressions régulières
* Manipulation de bases de données
* …

### Les bibliothèques tierces

* Identification et installation
* Découvertes de quelques bibliothèques incontournables (téléchargement de ressources internet, manipulation de données…)

**TP : réalisation d’un mini-projet permettant de manipuler plusieurs bibliothèques**