Python, perfectionnement

Référence : **DELY200** Durée : **3 jours** Certification : **Aucune**

CONNAISSANCES PREALABLES

• Avoir suivi le cours DEPYT001 - Python par la pratique ou posséder les connaissances et compétences équivalentes.

PROFIL DES STAGIAIRES

• Développeurs, administrateurs et architectes.

OBJECTIFS

• Décrire les subtilités du langage Python et en tirer parti pour écrire des programmes bien structurés, robustes et efficaces. • Gérer le développement en langage Python, de façon approfondie.

CERTIFICATION PREPAREE

Aucune

METHODES PEDAGOGIQUES

- Mise à disposition d'un poste de travail par stagiaire
- Remise d'une documentation pédagogique numérique pendant le stage
- La formation est constituée d'apports théoriques, d'exercices pratiques, de réflexions et de retours d'expérience
- Le suivi de cette formation donne lieu à la signature d'une feuille d'émargement

FORMATEUR

Consultant-Formateur expert Développement Internet

METHODE D'EVALUATION DES ACQUIS

- Auto-évaluation des acquis par le stagiaire via un questionnaire
- Attestation des compétences acquises envoyée au stagiaire
- Attestation de fin de stage adressée avec la facture

CONTENU DU COURS

Jour 1

Langage

- Appel de fonctions aspects avancés : *args, **argk
- Lambda, filter et map
- Utilisation avancée des modèles de données : list, dic, stack, queue
- Utilisation avancée des fonctions : passages d'arguments
- Aspects avancés de la Programmation Orientée Objets (POO)
- ***Exemples de travaux pratiques : Création d'un programme avec exploitation avancée des collections de Python ; Changement des types de passage d'argument***

Programmation multithread

- Concepts de bases : programme, thread, synchronisation
- Gestion de threads : modules thread, threading
- Threads et la Programmation Orientée Objets
- Gestion des aspects concurrentiels : lock, mutex, sémaphores...
- Threads et échanges de données
- Notion de pool de threads
- ***Exemples de travaux pratiques : Création d'un programme lançant plusieurs threads ; Synchronisation de ces threads pour obtenir un résultat dépendant de traitements parallèles ; Protection des données globales du programme écrit grâce aux mutex et lock ; Ajout d'un sémaphore pour l'accès à des ressources en nombre restreint ; Echange de données entre threads lancés ; Travail sur la notion de pool***

Programmation réseau avec les sockets

- Rappels sur le TCP/IP et concepts de base de l'API socket

- Utilisation du module socket
- Socket en mode connecté : TCP ou stream
- Socket en mode non connecté : UDP ou datagram
- Les sockets et la Programmation Orientée Objets
- Combinaison des sockets et des threads
- ***Exemples de travaux pratiques : Création d'un programme serveur puis client échangeant des données via les sockets en TCP puis UDP ; Démonstration de l'avantage du multi-threading dans ce cas pour les serveurs***

Jour 2

Python et XML

- Concepts de base : DOM (Document Object Model) ; SAX (Simple API for XML) ; Parser...
- Gestion de fichiers XML selon SAX et selon DOM
- Requêtage Xpath et transformation avec XSL
- ***Exemples de travaux pratiques : Création d'un programme de lecture d'un flux de données de taille importante via SAX ; Mise à jour d'une structure via DOM***

Programmation graphique

- Différentes API : Tkinter, wxPython, Qt/UI API
- Tkinter : présentation et mise en oeuvre
- Présentation et mise en oeuvre : API wxPython ; API Qt/UI
- ***Exemples de travaux pratiques : Ecriture d'un programme utilisant Qt/UI d'échange utilisateur avec l'interface graphique***

Persistance de données

- Concepts de base : sérialisation / désérialisation
- Différents modèles de persistance : Pickle...
- Persistance texte avec JSON et XML

Les bases de données

- Concepts de base : SQL, NoSQL, tables, curseur
- Création d'une base avec les modules Gadfly
- Gestion de la base de données SQLite et MySQL
- ***Exemples de travaux pratiques : Création d'un programme qui sérialise un flux JSON ; Ecriture d'un programme d'accès à une base de données MySQL ; Mise au point de requêtes de lecture, inertion, mise à jour depuis Python***

Jour 3

Intégration Python/C et Python/Java

- Présentation générale et mise en oeuvre de SWIG
- Python/C et les packages : Natifs C ; Natifs Java
- ***Exemple de travaux pratiques : Création d'un programme interfaçant avec des API écrites en C et en Java***

Mise au point de programme

- Débogage : exécution pas à pas
- Modes : verbose et trace
- Analyse des performances et profiling

*Notre **référent handicap** se tient à votre disposition au 01.71.19.70.30 ou par mail à referent.handicap@edugroupe.com pour recueillir vos éventuels besoins d'aménagements, afin de vous offrir la meilleure expérience possible.*