

1

# Python 3 – Présentation



# Plan de cours

Les bases de python

Dossiers, Fichiers et Librairies avec python

Outils de data analyse : Numpi, pandas, sckit learn, matplotlib

**Python pour l'administration**

Django et les API

# Gestion de système avec Python

- ▮ Le métier de développeur/devsops en 2023
  - ▮ L'assemblage de composant
  - ▮ Github copilot
  - ▮ ChatGPT
  
- ▮ Présentation de librairie pour réaliser des applications avancées d'administration
  - ▮ Interface graphique (principe, exemple)
  - ▮ Cryptage (principe, exemple)
  - ▮ Connection SSH et commandes sur un serveur
  - ▮ Transfert FTP (principe, exemple)
  - ▮ Gestion des emails (principe, exemple)

# Assemblage de composant

- ↩ Le métier de développeur : de création de code vers assemblage des composants
  - ↩ Design pattern : ensemble de code standard pour réaliser certaines fonctionnalités
  - ↩ Framework : bibliothèque clé en main (Symfony, jQuery, kivy, ...)
- ↩ Python « langage Glue »
  - ↩ Permet de se connecter et interagir avec d'autres applications (API)
  - ↩ Facilité d'intégrer différents systèmes et sources de données
  - ↩ Capacité à développer des applications complexes
  - ↩ Permet de relier plusieurs composants
  - ↩ Nombreuses bibliothèques standard, tierces, que l'on pouvait créer soi-même
- ↩ Dans le cadre du métier de DevOps
  - ↩ Accéder à de la donnée (DB, fichier, logs, api, ...)
  - ↩ Administration et supervision d'applications tierces (ERP, site, ...)
  - ↩ Mise en place d'architecture (matériel, OS, conteneur, ...)

# Github copilot

- Outil d'aide au développement
  - 2018 : rachat de Github par Microsoft
  - 2021 : présentation de la technologie et première version
  - 2022 : sortie de l'extension dans visual studio code (sur liste d'attente)
  - Juin 2022 : première version stable disponible par abonnement (10€)
  - Complète le code à partir de commentaire ou appel de fonction
  - IA qui se base sur les sources présentes dans sur les dépôts Git
  - Fiable à 50% mais permet d'avoir des pistes (syndrome de la page blanche)
- 
- Rentabilité de prendre un abonnement (taux horaire)
  - Calcul mental vs calculatrice

# Chat GPT

- Présentation au public fin 2022
- Permet d'avoir une base travail pour une première version de code
- Evite le syndrome de la page blanche
- Permet de corriger des sources (debugging)
- Pas mal d'erreur aussi
- Limite : technologie trop récente (API notion fin 2022)

# Interface Graphique avec kivy

- ▮ Usage : rendre plus simple la manipulation d'une application
- ▮ Utilisation d'une librairie (tkinter, kivy) pour créer des fenêtres graphiques
- ▮ Crée une interface multiplateforme (windows/linux/mac)
- ▮ Possibilité de créer un package android avec buildozer (kivy)
  
- ▮ Création d'une fenêtre principale avec un grillage
  - ▮ Ajout d'éléments (bouton, zone de texte, ...) sur la fenêtre
- ▮ Programmation événementielle
  - ▮ L'appuie sur un bouton déclenche un événement qui appelle un traitement
  - ▮ La fenêtre reste ouverte sur l'écran

# Architecture d'un programme kivy



# Cryptage

- ▮ Cryptage de base (librairie « cryptography ») => pip install cryptography
  - ▮ symétrique : une seule clé pour crypter et décrypter un message
    - ▮ Si on intercepte la clé il est possible de décrypter tous les messages
  - ▮ asymétrique : utilisation d'une clé publique et d'une autre privée
    - ▮ La clé publique sert à crypter les messages
    - ▮ La clé privée sert à décrypter les messages
- ▮ Hashage de mot de passe (librairie « hashlib ») => pip install hashlib
  - ▮ Permet de conserver un mot de passe en base de données
  - ▮ Création d'une clé MD5 pour l'inaltération d'un document

# Connection ssh

- ▮ Utilisation de la bibliothèque **paramiko**
- ▮ Permet de se connecter sur un serveur et passer des commandes sur celui-ci
- ▮ Il est possible de s'identifier par login/password classique mais aussi par clé
- ▮ L'exécution de commande retourne le
  - ▮ stdin : saisie d'informations complémentaires
  - ▮ Stdout : récupère la réponse de la commande lancée
  - ▮ Stderr : les erreurs de la commande lancée si il y en a

# Utilisation de FTP (File Transfert Protocol)

- ▢ Usage : transfert de fichiers entre deux machines distantes
- ▢ Il est possible de créer avec python un serveur et un client FTP
- ▢ Serveur FTP : `pip install pyftplib`
  - ▢ Il est possible d'installer un serveur ftp sur sa propre machine
  - ▢ Mise à disposition d'un espace de stockage distant
  - ▢ Gestion des utilisateurs (identifié ou non) pouvant accéder à l'espace de stockage
  - ▢ Trace des accès distant et des commandes passées
- ▢ Client FTP : `pip install ftplib` (  
<https://docs.python.org/3/library/ftplib.html>)
  - ▢ Permet d'accès au serveur FTP
  - ▢ Transfert vers et depuis des fichiers sur le serveur FTP
  - ▢ Manipulation des dossiers et des fichiers à distance

# Client Email

- ▮ Usage : permettre à une application de gérer des emails
  - ▮ Emission : Transmettre un email en cas d'anomalie sur une machine
  - ▮ Transmission : Déclencher un traitement sur une machine à la réception d'un email
- ▮ Principe émission :
  - ▮ Se connecter à un serveur SMTP (Simple Mail Transfert Protocol)
  - ▮ Créer un email (émetteur, destinataire, sujet, message, option : pièce-jointe)
  - ▮ Transmettre un email
- ▮ Principe réception :
  - ▮ Se connecter au serveur (IMAP ou POP3) de stockage des emails
    - ▮ IMAP : va conserver les messages sur le serveur
    - ▮ POP3 : effacera au bout d'un certain temps les messages du serveur
  - ▮ Récupération des messages non lus
  - ▮ Analyse du contenu d'un email (émetteur, destinataire, sujet, message, option : pièce-jointe)
  - ▮ Déclencher un traitement (message box)