Langages informatiques – Python

Université Panthéon-Sorbonne – EMS

Enseignant : Matthieu Jacq - matthieu.jacq@univ-paris1.fr

Objectifs du cours

- Préparer la certification TOSA
- Savoir utiliser un IDE (VSCode)
- Être autonome pour comprendre et écrire des scripts en python
  - Maîtriser la syntaxe de base : variables, boucles, conditions, fonctions, classes
  - Connaître les fonctions de base de la bibliothèque standard

Objectif de la séance

- Présentation de python
- Modalités d'évaluation et de certification
- Installation de l'environnement de travail : VSCode (et extenstions), git (et compte github), python
- Savoir utiliser un terminal avec bash ou zsh
- Premières manipulations d'un notebook jupyter dans VSCode

Pourquoi python ?

De multiples raisons :

- Langage très populaire
- Facile à apprendre
- Très polyvalent
- Très utilisé pour le traitement de données et l'IA

Pourquoi python ?

Comparaison avec d'autres langages

- Python:

```python
print("Hello, World!")
```

- javascript:

```javascript
console.log("Hello, World!");
```

Pourquoi python ?

Comparaison avec d'autres langages

- Python:

```
print("Hello, World!")
```

- C:

```
#include <stdio.h>
int main() {
    printf("Hello, World!\n");
    return 0;
}
```

Pourquoi python ?

Comparaison avec d'autres langages

- Python:

```python
print("Hello, World!")
```

- Java:

```java
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

Évaluation et certification

	Evaluation	Certification
Nombre de questions	25	35
Durée du test	60min	90min
Modalités de passage	En ligne	Passage en conditions d'examen dans un centre agréé ou en ligne (e-surveillance)
Résultats	Niveau sur une échelle de 1 à 5. Délivrance immédiate du rapport détaillé de compétences	Score sur une échelle de 1 à 1000. Délivrance de la certification si le score est supérieur à 551 sous 5 jours ouvrés

Évaluation et certification

Échelle de notation

Niveau	Scores Tosa®
Expert	876 - 1000
Avancé	726 – 875
Opérationnel	551 – 725
Basique	351 – 550
Initial	1 – 350

Référentiel de compétences

Pourquoi utiliser VSCode

- Gratuit et open-source
- Léger et rapide
- Très personnalisable avec des extensions
- Supporte de nombreux langages de programmation
- Intégration avec git et GitHub
- Multi-plateforme (Windows, MacOS, linux)
- Utilisé par de nombreux développeurs professionnels
- Vous permettra d'écrire et d'exécuter du code python sur votre machine sans être dépendant d'une solution commerciale en ligne

Installation de VSCode

- Télécharger et installer VSCode

> Si lors de l'installation, on vous propose d'installer github copilot, je vous déconseille de le faire pour l'instant. C'est un outil d'autocomplétion basé sur de l'IA qui est très intéressant, mais qui vous empêchera de faire correctement vos exercices et de progresser.

🎁 Pourquoi créer un compte GitHub et installer git

- GitHub est une plateforme de gestion de code source très populaire
- Permet de sauvegarder et de partager son code
- Permet de collaborer avec d'autres développeurs
- Permet de suivre l'historique des modifications du code
- Git est un logiciel de gestion de versions du code source
- Permet de travailler en local sur son ordinateur et de synchroniser son code avec GitHub

🎁 Création d'un compte GitHub

- Aller sur GitHub et créer un compte ou se connecter à un compte existant

Installation de git

> git est un logiciel de gestion de versions du code source

- **Windows**: Télécharger et installer git et `git bash` (inclus dans l'installation de git)
- 🎁**MacOS/linux** Vérifier l'installation en ouvrant un terminal et en tapant `git --version`

> 🎁 Sous macos, il sera nécessaire d'installer les outils en ligne de commande Xcode (Xcode Command Line Tools) si ce n'est pas déjà fait. Pour cela, taper `xcode-select --install` dans un terminal. De plus `bash` ou son équivalent `zsh` est déjà installé par défaut. Par conséquent il est facultatif d'installer git.

Vérification du terminal

1️⃣ Windows :

Une fois que git est installé sur windows :

- Afficher le panneau du terminal dans VS Code (`View` -> `Terminal`)
- Cliquer sur la petite flèche 🔽 à côté de l'icône `+` en haut à droite du panneau du terminal

- Sélectionner `Git Bash` dans la liste déroulante
- Vous pouvez enfin taper `git --version` dans le panneau pour vérifier l'installation.

Vérification du terminal

2 macOS / linux :

- Afficher le panneau du terminal dans VS Code (`View` -> `Terminal`)
- Vous pouvez laisser le terminal par défaut (bash ou zsh)
- Vous pouvez entrer n'importe quelle commande bash dans le panneau du terminal, par exemple `echo hello` qui affichera `hello` dans le terminal.

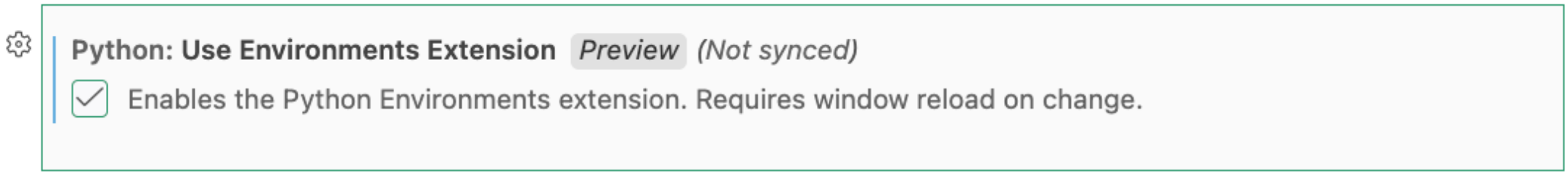
Installation de Python

- Télécharger et installer Python (installer la version standalone pas le manager python)
  - ⚠️ **Windows**: Bien cocher "Add to path" lors de l'installation

ℹ️ On peut vérifier l'installation en ouvrant un terminal (par exemple celui de VSCode) et en tapant `python --version` (ou `python3 --version`)

Extensions à installer dans VSCode

- Python (Microsoft): apporte de la coloration syntaxique et des outils pour python (autocompletion, linting, debugging, etc.)
- Python Environments (Microsoft): permet de gérer les environnements virtuels
- Jupyter (Microsoft): permet de créer des notebooks jupyter

> Pour Python Environments, il faudra l'activer manuellement dans les paramètres de VSCode (`File` -> `Preferences` -> `Settings` -> Rechercher `Python Environments` -> cocher `Python: Enable Env Management`)
>
> 
>
>
> Il faudra ensuite redémarrer VSCode pour que l'extension soit prise en compte.

Les fichiers python

- Les fichiers python ont l'extension `.py`
- On peut les exécuter en tapant `python nom_du_fichier.py` dans un terminal
- On peut aussi les exécuter dans VSCode en cliquant sur le bouton "Run" en haut à droite

Ouvrez le fichier `main.py` sur l'EPI et exécutez-le. Que se passe-t-il ?

Qu'est-ce qu'un terminal ?

- Un terminal est une interface en ligne de commande qui permet d'interagir avec le système d'exploitation.

  > Lorsque je donnerais des instructions pour taper des commandes, nous utiliserons l'interprêteur de commandes `bash` (disponible par défaut sur macOS et linux et disponible sur windows après l'installation de git).

- On peut ouvrir un terminal intégré dans VSCode en cliquant sur `Terminal` -> `New Terminal`
- On peut exécuter des commandes en tapant dans le terminal
- On peut exécuter des scripts python en tapant `python nom_du_fichier.py` dans le terminal

> 👉 Essayez de taper `python main.py` dans le terminal intégré de VSCode.

Les tests unitaires

Les tests unitaires permettent de vérifier que le code fonctionne correctement

Installer le package `pytest` depuis l'onglet python dans vscode (ou en tapant `pip install pytest` dans un terminal), puis ouvrez le fichier `test_main.py`.

Les notebooks Jupyter

Les notebooks Jupyter permettent de mélanger du code, du texte et des images.

Cela constitue un outil très puissant pour l'analyse de données et la présentation de résultats.

> Les notebooks jupyter seront notre principal outil de travail pour le cours.

Ouvrez le fichier `notebook.ipynb` pour la suite du cours.

Les notebooks Jupyter : sélection d'un interpréteur (kernel)

- Un notebook jupyter doit être associé à un interpréteur python (kernel) pour pouvoir exécuter du code
- Pour sélectionner un kernel, cliquer sur le bouton en haut à droite (à côté de "Python 3") et choisir l'interpréteur python installé précédemment
- Si l'interpréteur n'apparaît pas, cliquer sur "Select Another Interpreter..." et choisir l'interpréteur python installé précédemment
- 👉 Essayez de sélectionner l'interpréteur python dans le notebook `notebook.ipynb`

Vous pouvez maintenant exécuter les cellules du notebook en cliquant sur le bouton "Run" (ou la flèche ▶️) à gauche de chaque cellule ou en appuyant sur `Shift+Enter`.

Qu'est-ce qu'un environnement virtuel ?

- Un environnement virtuel est un espace isolé qui permet d'installer des packages python sans interférer avec les autres projets
- Permet de gérer les dépendances d'un projet python
- Permet de garantir que le code fonctionne de la même manière sur différentes machines
- Permet de tester différentes versions de packages sans conflit

Qu'est-ce qu'une librairie python ?

- Une librairie python est un ensemble de modules python qui fournissent des fonctionnalités supplémentaires
- Permet de réutiliser du code existant
- Permet de gagner du temps en évitant de réinventer la roue
- Exemples de librairies python populaires :
  - NumPy : calcul scientifique
  - Pandas : manipulation de données
  - Matplotlib : visualisation de données

Comment installer une librairie python ?

1 Méthode 1 : Utiliser pip

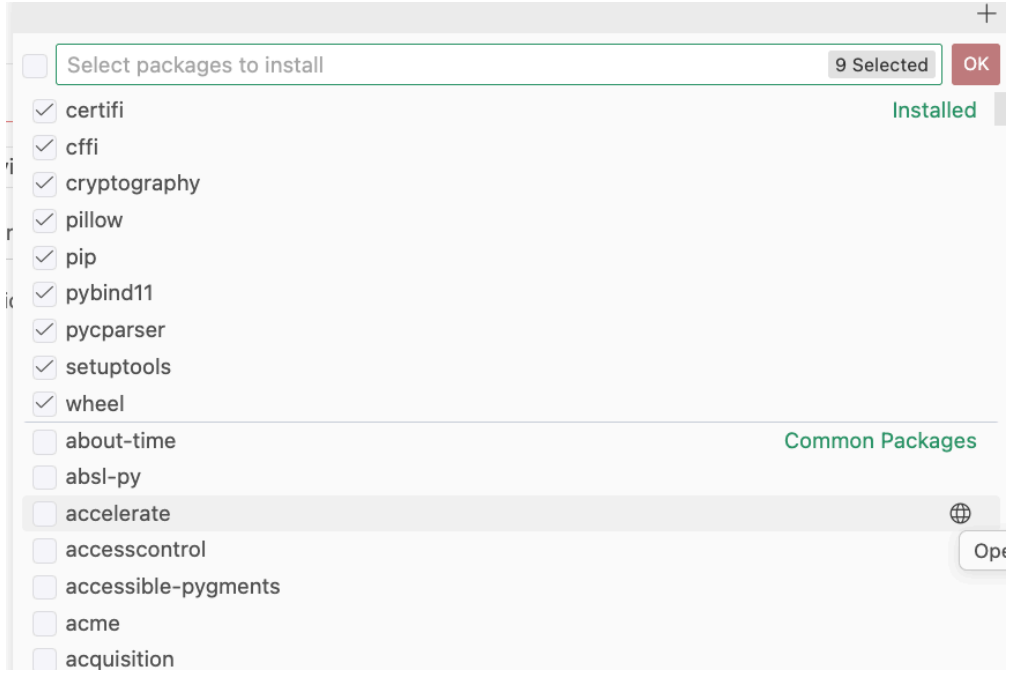
- On peut installer une librairie python en utilisant le gestionnaire de packages `pip`
- La commande pour installer une librairie est `pip install nom_de_la_librairie` (dans un terminal)

Comment installer une librairie python ?

2 Méthode 2 : Utiliser un gestionnaire de packages

Nous allons utiliser le gestionnaire de packages intégré de VSCode

- Ouvrir l'onglet python dans VSCode (icône de serpent dans la barre latérale gauche si l'extension Python Environments est installée)
- Cliquer sur une l'icône de package à côté de l'environnement python que vous souhaitez utiliser
- Rechercher la librairie à installer et cocher la case correspondante pour l'installer



Les bases de VSCode

- 📂 Ouvrir un dossier : `File` -> `Open Folder...`
- 📄 Ouvrir un fichier : `File` -> `Open File...`
- ✏️ Éditer un fichier : cliquer sur le fichier dans l'explorateur de fichiers à gauche
- 💾 Sauvegarder le fichier : `File` -> `Save` (ou `Ctrl+S` sur Windows/Linux, `Cmd+S` sur MacOS)
  - ⚠️ si vous modifiez un fichier, un indicateur `•` apparaissant dans l'onglet du fichier pour prévenir que vous avez des modifications non sauvegardées
- 🗑️ Fermer le fichier : `File` -> `Close` (ou `Ctrl+W` sur Windows/Linux, `Cmd+W` sur MacOS)
- 🔍 Rechercher dans le fichier : `Edit` -> `Find` (ou `Ctrl+F` sur Windows/Linux, `Cmd+F` sur MacOS)

Les bases de VSCode

- ➕ Ouvrir une nouvelle fenêtre : `File` -> `New Window` (ou `Ctrl+Shift+N` sur Windows/Linux, `Cmd+Shift+N` sur MacOS)
- 💻 Ouvrir un terminal intégré : `Terminal` -> `New Terminal`
- ▶️ Ouvrir la palette de commandes : `View` -> `Command Palette...` (ou `Ctrl+Shift+P` sur Windows/Linux, `Cmd+Shift+P` sur MacOS)
- 📱 Installer des extensions : `View` -> `Extensions`
- 🔧 Ouvrir les paramètres : `File` -> `Preferences` -> `Settings` (ou `Ctrl+,` sur Windows/Linux, `Cmd+,` sur MacOS)