

IUT Colmar

SAE 3.02

Documentation d'installation et d'utilisation

Marius Keltz
02/01/2025

Sommaire :

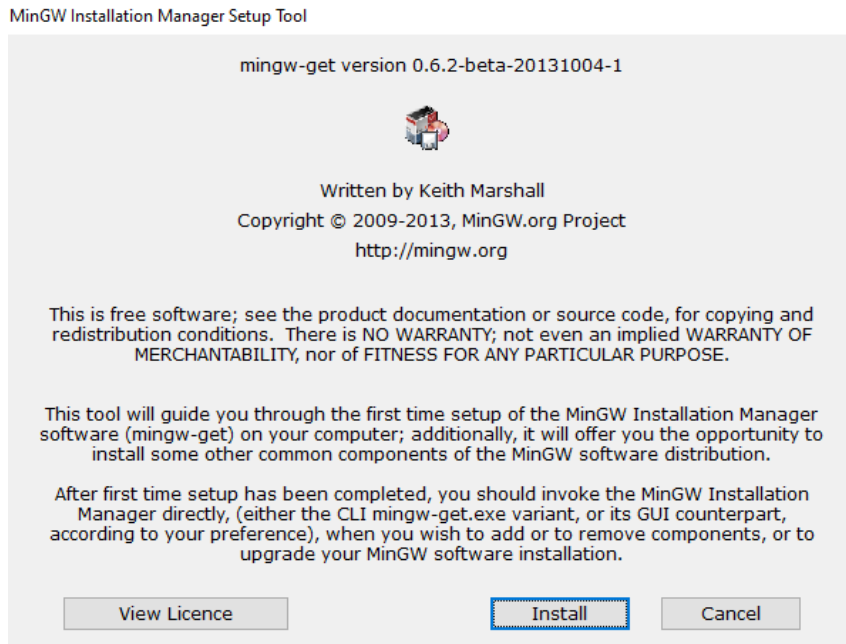
Sommaire :	1
Windows :	2
Installation Compilateur C :	2
Installation Compilateur Java :	5
Installation Compilateur Python :	6
VSCode :	7
Vérification :	8
Programme en C :	8
Programme en Python :	8
Programme en Java :	8
Lancement du projet :	9
Linux :	11
Installation Compilateur Python :	11
Installation Compilateur C :	11
Installation Compilateur Java :	11
Installation Interface Graphique :	12
Problème de fermeture de Serveur :	12
Lancement du projet :	12

Windows :

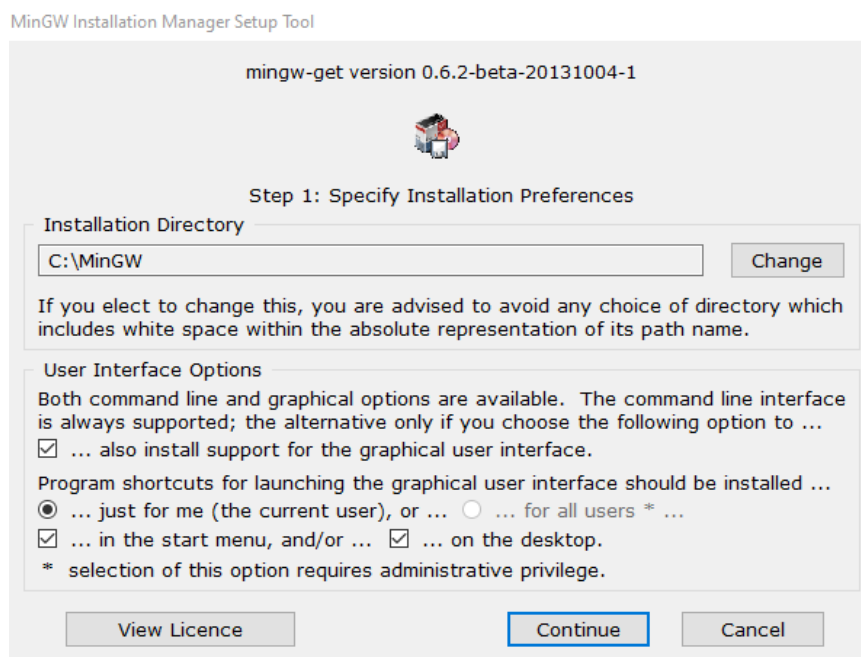
Installation Compilateur C :

On va devoir installer un compilateur GCC pour pouvoir exécuter des programmes en C, pour ce faire nous allons installer [mingw](http://mingw.org).

Une fois avoir installé mingw-setup.exe sur le site précédent, lancez-le.



Une fenêtre d'installation va se lancer, cliquez sur Install.



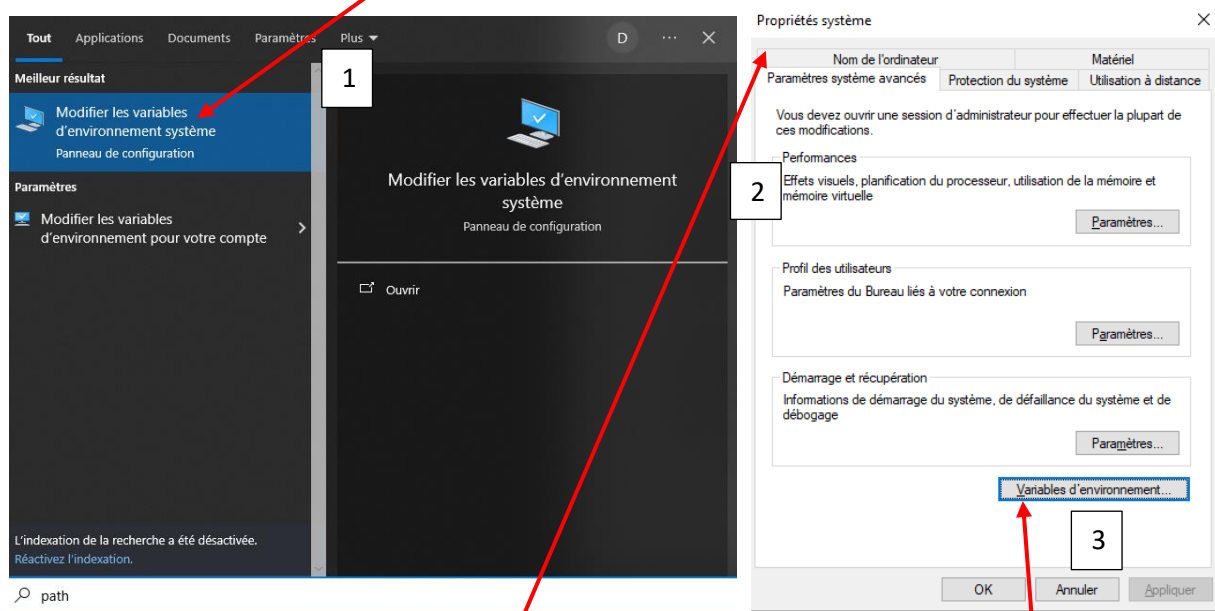
Il vous demande si vous voulez un chemin spécifique, pour notre part nous allons tout laisser par défaut, puis Mingw va s'installer. Enfin cliquez sur continuer

Nous avons choisi d'installer mingw de façon graphique donc une fenêtre se lance et nous demande de choisir ce que nous voulons installer :

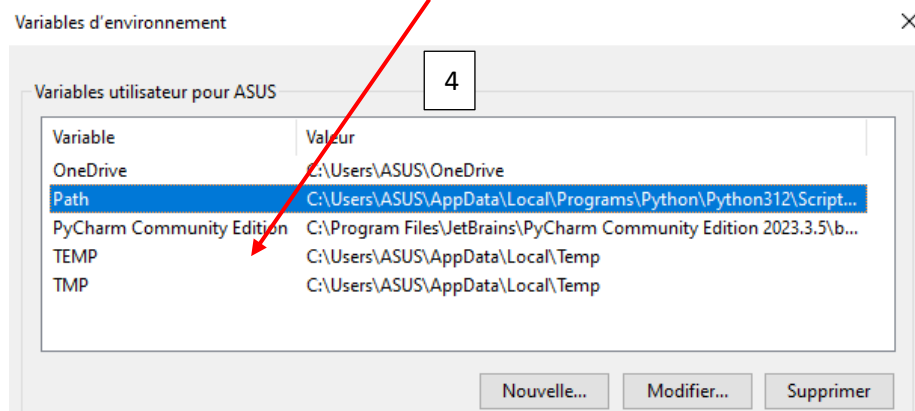
Package	Class	Installed Version	Repository Version	Description
<input type="checkbox"/> mingw-developer-tool...	bin		2013072300	An MSYS Installation for MinGW Developers (meta)
<input type="checkbox"/> mingw32-base	bin		2013072200	A Basic MinGW Installation
<input type="checkbox"/> mingw32-gcc-ada	bin		6.3.0-1	The GNU Ada Compiler
<input type="checkbox"/> mingw32-gcc-fortran	bin		6.3.0-1	The GNU FORTRAN Compiler
<input checked="" type="checkbox"/> mingw32-gcc-g++	bin	6.3.0-1	6.3.0-1	The GNU C++ Compiler
<input checked="" type="checkbox"/> mingw32-gcc-objc	bin	6.3.0-1	6.3.0-1	The GNU Objective-C Compiler
<input type="checkbox"/> msys-base	bin		2013072300	A Basic MSYS Installation (meta)

Nous allons installer mingw32-gcc-g++ et mingw32-gcc-objc. Une fois avoir sélectionné les deux compilateurs, cliquez en haut à gauche sur Installation puis sur « Apply Change ». Il va vous demander de confirmer l'installation, cliquez sur Apply.

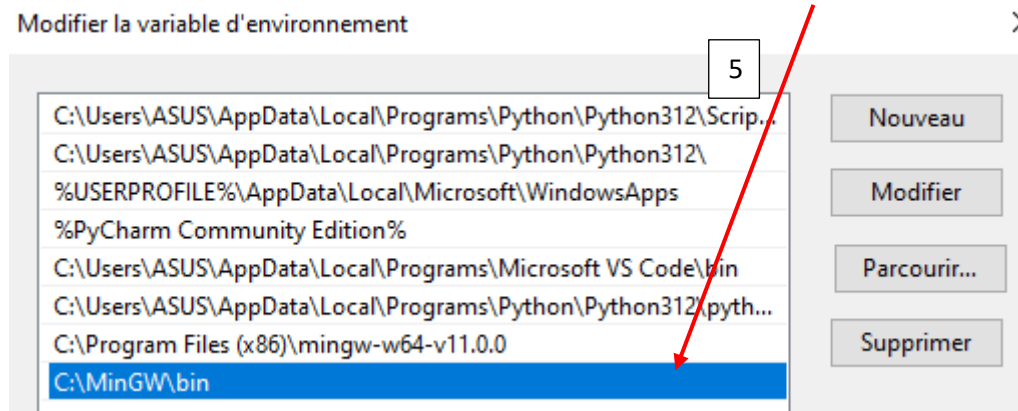
Maintenant que notre compilateur C est installé, il faut le rajouter dans le PATH. Cliquez sur le bouton Windows en bas à droite et cherchez PATH :



Vous vous retrouvez dans les propriétés du système d'exploitation de Windows. Cliquez sur Variables d'environnement. Une fois dans les variables d'environnement, allez dans la variable PATH qui se situe dans les variables de votre utilisateur.



Une fois dans la variable PATH, nous allons ajouter Mingw. Si vous n'avez pas changé l'emplacement d'installer de MinGW, il se situe dans votre disque C:\MinGW\bin. Si vous avez changé l'emplacement, faite attention à bien sélectionner le dossier bin qui se situe dans MinGW.



Voilà, l'installation du compilateur python est terminé, il n'y a pas besoin de le rajouter au PATH.

Installation Compilateur Java :

Pour que le serveur puisse renvoyer la réponse au client, il va bien falloir installer un compilateur java. Nous allons nous rendre sur le site officiel de [Java](#) pour l'installer.

Une fois avoir téléchargé JavaSetup.exe sur le site officiel, on va le lancer. Il nous demande si nous voulons changer l'emplacement d'installation, pour ma part je vais laisser celui par défaut.



Cliquer sur Installer et l'installation va se faire toute seule.

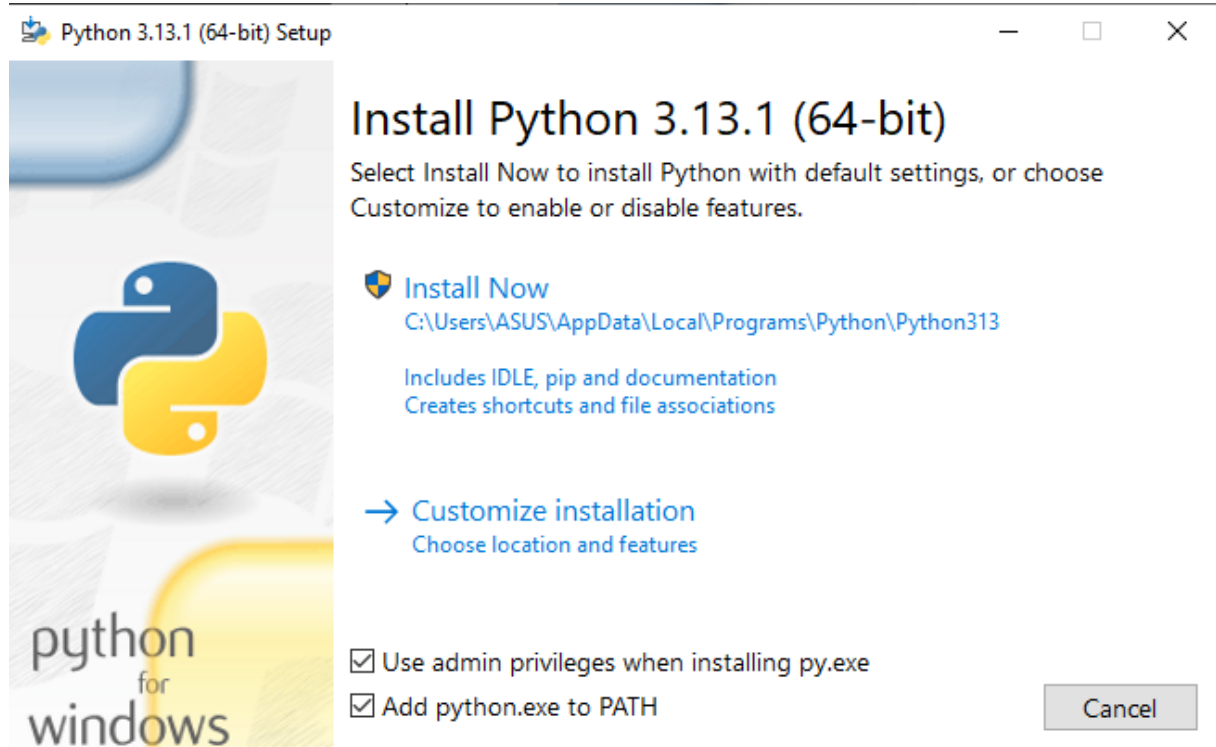


Voilà le compilateur Java est installé, il n'y a pas besoin de le rajouter au PATH.

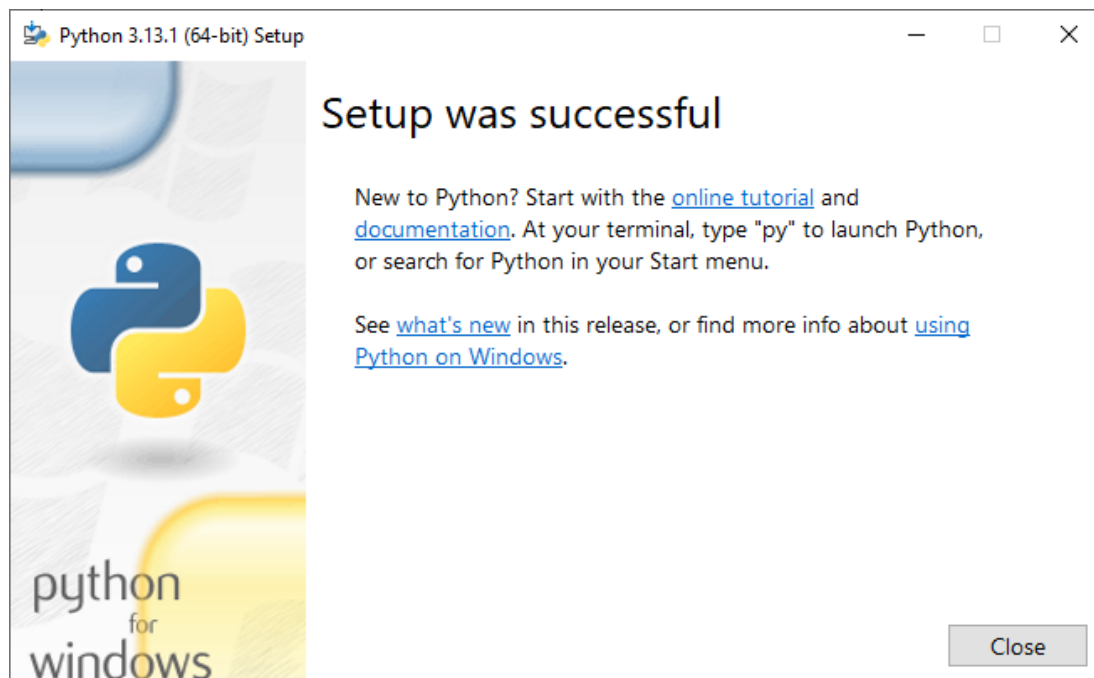
Installation Compilateur Python :

Dans le cadre de la SAE nous avons comme condition d'utiliser Python 3, donc nous allons aller sur le site officiel de [Python](https://python.org) pour l'installer. Pour ma part, j'ai pris la dernière version python 3.13.1.

Une fois le fichier python 3-13.1 téléchargé, nous allons le lancer pour démarrer l'installation du compilateur.



Pensez à cocher le **Add Python.exe to PATH** ce qui nous permettra de ne pas le faire par la suite et de gagner du temps. Une fois ceci fait, cliquez sur Install Now.

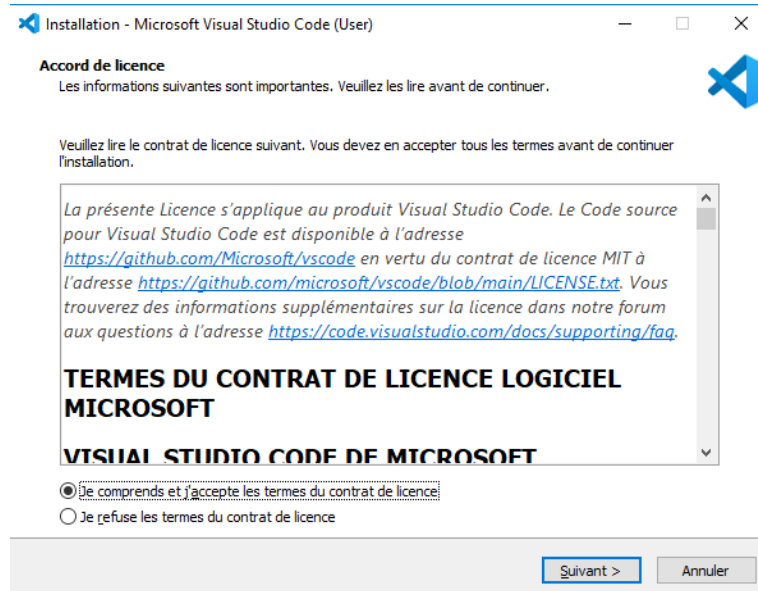


Voilà l'installation du compilateur python est terminée, il n'y a pas besoin de le rajouter au PATH.

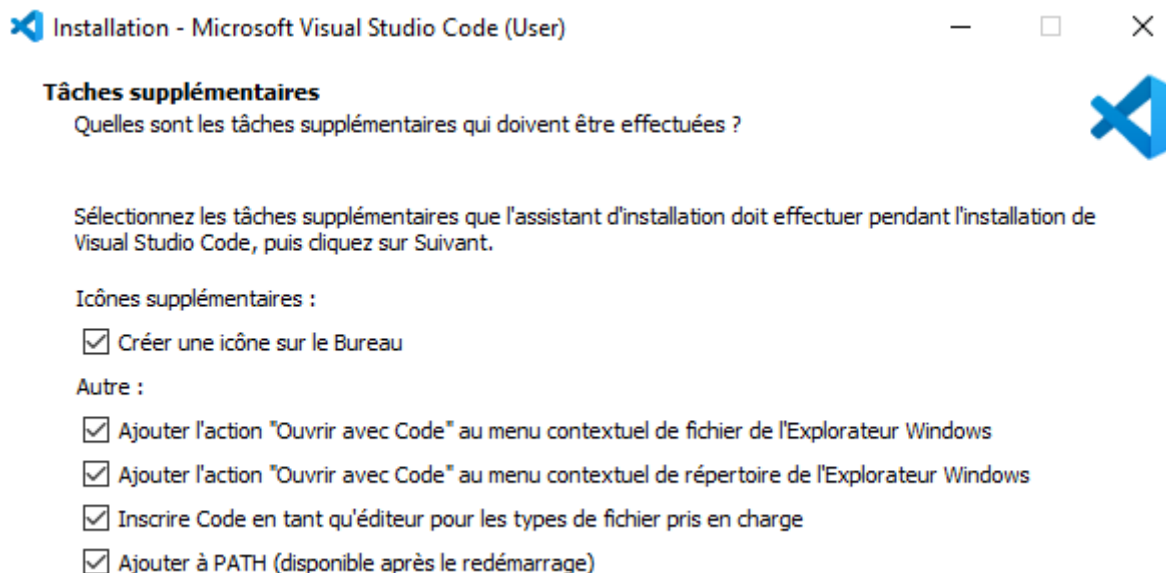
VSCode :

En ce qui me concerne, j'ai utilisé comme outil de programmation VSCode, que vous pouvez l'installer sur le site officiel de [Visual Studio](https://code.visualstudio.com/).

Une fois VSCode téléchargé sur le site, on va le lancer pour démarrer l'installation du logiciel.



Acceptez le contrat de Licence, puis cliquez sur suivant. Il vous propose de choisir l'emplacement d'installation, pour ma part j'ai laissé par défaut.



Pensez à ajouter VSCode au Path, puis installer VSCode. Voilà, maintenant vous avez fini l'installation de VSCode et vous pouvez l'utiliser.

Vérification :

Nous allons vérifier si les différents compilateurs fonctionnent, copiez les codes ci-dessous et regardez si vous n'avez pas d'erreurs.

Programme en C :

```
#include <stdio.h>
int main() { printf("Bonjour depuis C !\n"); return 0; }
```

```
Microsoft Windows [version 10.0.19045.4170]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\ASUS>cd c:\Users\ASUS\GitHub\CloneRepertory\Cluster-de-Calculs\
c:\Users\ASUS\GitHub\CloneRepertory\Cluster-de-Calculs>gcc C.c -o C
c:\Users\ASUS\GitHub\CloneRepertory\Cluster-de-Calculs>C.exe
Bonjour depuis C !
```

Programme en Python :

```
import time
time.sleep(1)
print("hello world")
```

```
c:\Users\ASUS\GitHub\CloneRepertory\Cluster-de-Calculs>python py.py
hello world
```

Programme en Java :

```
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello from Java!");
    }
}
```

```
c:\Users\ASUS\GitHub\CloneRepertory\Cluster-de-Calculs>java java.java
Hello from Java!
```

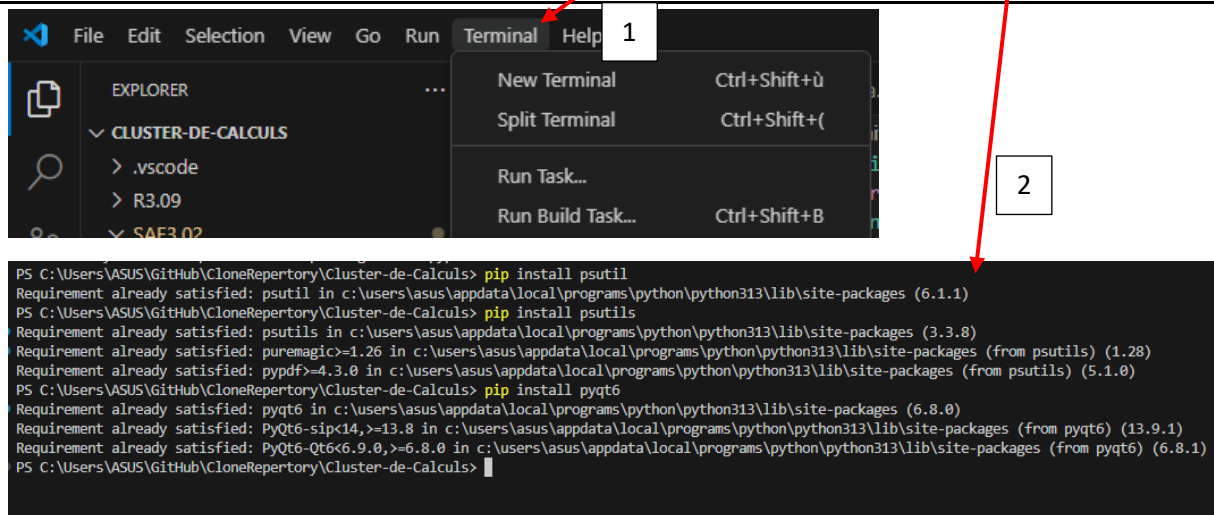
Voilà les résultats que vous devez avoir lorsque vous exécutez les programmes ci-dessus.

Lancement du projet :

Pour faire fonctionner le serveur maître, serveur esclave ou encore le client, il faut récupérer le projet sur le [github](#), puis lancer le dossier avec VSCode.

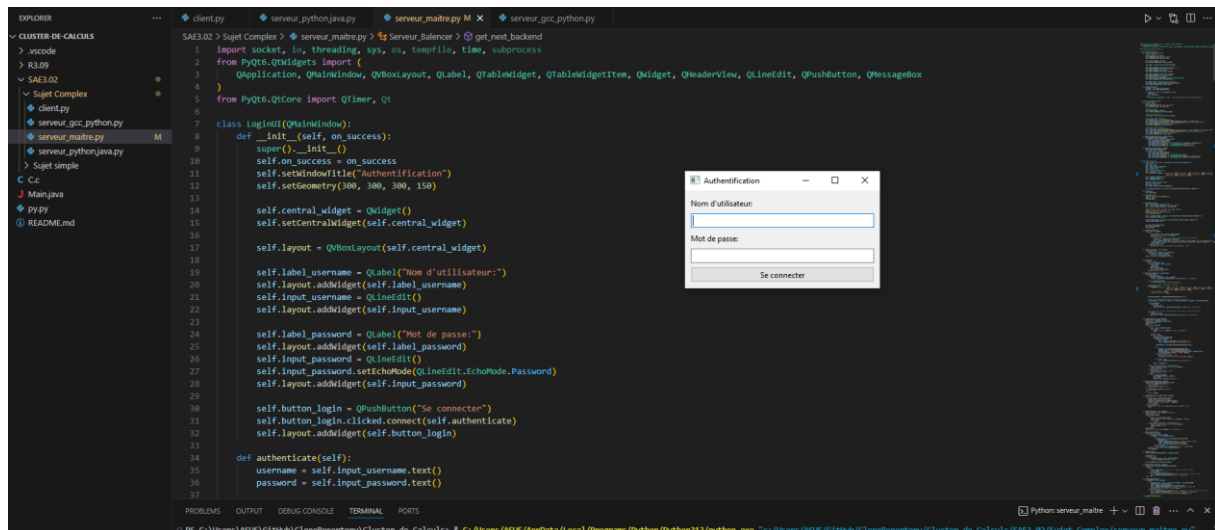
Il faut installer les dépendances, il vous suffit d'ouvrir un nouveau terminal et d'exécuter les commandes suivantes :

```
pip install pyqt6
pip install psutils
pip install psutil
```



```
PS C:\Users\ASUS\GitHub\CloneRepertory\Cluster-de-Calculs> pip install psutil
Requirement already satisfied: psutil in c:\users\asus\appdata\local\programs\python\python313\lib\site-packages (6.1.1)
PS C:\Users\ASUS\GitHub\CloneRepertory\Cluster-de-Calculs> pip install psutils
Requirement already satisfied: psutils in c:\users\asus\appdata\local\programs\python\python313\lib\site-packages (3.3.8)
Requirement already satisfied: puremagic>=1.26 in c:\users\asus\appdata\local\programs\python\python313\lib\site-packages (from psutils) (1.28)
Requirement already satisfied: pypdf>=4.3.0 in c:\users\asus\appdata\local\programs\python\python313\lib\site-packages (from psutils) (5.1.0)
PS C:\Users\ASUS\GitHub\CloneRepertory\Cluster-de-Calculs> pip install pyqt6
Requirement already satisfied: pyqt6 in c:\users\asus\appdata\local\programs\python\python313\lib\site-packages (6.8.0)
Requirement already satisfied: PyQt6-sip<14,>=13.8 in c:\users\asus\appdata\local\programs\python\python313\lib\site-packages (from pyqt6) (13.9.1)
Requirement already satisfied: PyQt6-Qt6<6.9.0,>=6.8.0 in c:\users\asus\appdata\local\programs\python\python313\lib\site-packages (from pyqt6) (6.8.1)
PS C:\Users\ASUS\GitHub\CloneRepertory\Cluster-de-Calculs>
```

Une fois ceci fait, vous pouvez lancer le serveur maître sans problème.



```
1 import socket, io, threading, sys, os, tempfile, time, subprocess
2 from PyQt6.QtWidgets import (
3     QApplication, QMainWindow, QVBoxLayout, QLabel, QTableWidgetItem, QTableWidgetItemEdit, QFileDialog, QHeaderView, QLineEdit, QPushButton, QMessageBox
4 )
5 from PyQt6.QtCore import QTimer, Qt
6
7 class LoginUI(QMainWindow):
8     def __init__(self, on_success):
9         super().__init__()
10         self.on_success = on_success
11         self.setWindowTitle("Authentication")
12         self.setGeometry(300, 300, 300, 150)
13
14         self.central_widget = QWidget()
15         self.setCentralWidget(self.central_widget)
16
17         self.layout = QVBoxLayout(self.central_widget)
18
19         self.label_username = QLabel("Nom d'utilisateur:")
20         self.layout.addWidget(self.label_username)
21         self.input_username = QLineEdit()
22         self.layout.addWidget(self.input_username)
23
24         self.label_password = QLabel("Mot de passe:")
25         self.layout.addWidget(self.label_password)
26         self.input_password = QLineEdit()
27         self.input_password.setEchoMode(QLineEdit.EchoMode.Password)
28         self.layout.addWidget(self.input_password)
29
30         self.button_login = QPushButton("Se connecter")
31         self.button_login.clicked.connect(self.authenticate)
32         self.layout.addWidget(self.button_login)
33
34     def authenticate(self):
35         username = self.input_username.text()
36         password = self.input_password.text()
```

On peut très bien le faire par ligne de commande sinon.

Serveur Esclave 1 :

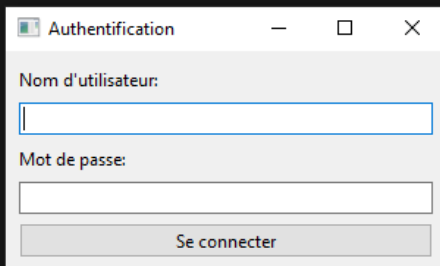
```
PS C:\Users\ASUS\GitHub\CloneRepertory\Cluster-de-Calculs> cd '.\SAE3.02\Sujet Complex\'
PS C:\Users\ASUS\GitHub\CloneRepertory\Cluster-de-Calculs\SAE3.02\Sujet Complex> python.exe .\serveur_gcc_python.py
Serveur esclave démarré sur 0.0.0.0:5002
En attente de la connexion du serveur maître...
```

Serveur Esclave 2 :

```
PS C:\Users\ASUS\GitHub\CloneRepertory\Cluster-de-Calculs> cd '.\SAE3.02\Sujet Complex\'
PS C:\Users\ASUS\GitHub\CloneRepertory\Cluster-de-Calculs\SAE3.02\Sujet Complex> python.exe .\serveur_python_java.py
Serveur esclave démarré sur 0.0.0.0:5001
En attente de connexions...
```

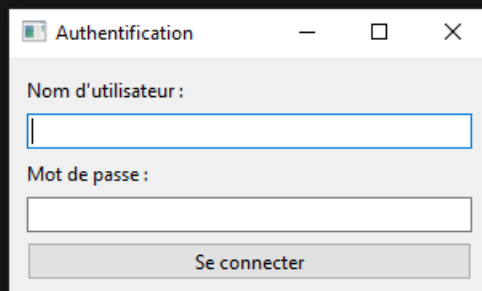
Serveur Maître :

```
PS C:\Users\ASUS\GitHub\CloneRepertory\Cluster-de-Calculs> cd '.\SAE3.02\'
PS C:\Users\ASUS\GitHub\CloneRepertory\Cluster-de-Calculs\SAE3.02> cd '.\Sujet Complex\'
PS C:\Users\ASUS\GitHub\CloneRepertory\Cluster-de-Calculs\SAE3.02\Sujet Complex> python.exe .\serveur_maitre.py
```

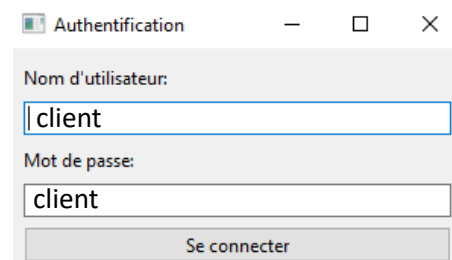
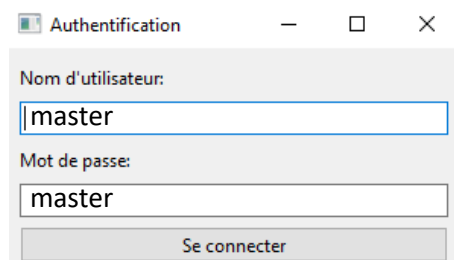


Client :

```
PS C:\Users\ASUS\GitHub\CloneRepertory\Cluster-de-Calculs> cd '.\SAE3.02\Sujet Complex\'
PS C:\Users\ASUS\GitHub\CloneRepertory\Cluster-de-Calculs\SAE3.02\Sujet Complex> python.exe .\client.py
```



Login Client et Serveur :



Linux :

Pour la partie Linux, j'ai fait une machine virtuelle sur une base Ubuntu. Premièrement, nous allons mettre à jour les paquets

```
sudo apt update  
sudo apt upgrade
```

Installation Compilateur Python :

Pour l'installation du compilateur python sur linux, nous allons toujours utiliser python3 :

Sur votre machine linux utilisez ces commandes :

```
sudo apt install python3 python3-venv python3-dev
```

Nous allons créer un environnement virtuel python :

```
python3 -m venv venv  
source venv/bin/activate
```

Ce qui va nous permettre d'installer les outils qui nous manquent pour le bon fonctionnement du projet tels que utils, et pyqt6. On va installer les dépendances qui nous manquent :

```
pip install pyqt6  
pip install psutils  
pip install psutil
```

Installation Compilateur C :

Pour l'installation du compilateur C sur linux, nous allons toujours utiliser gcc :

```
sudo apt update  
sudo apt install gcc openjdk-11-jdk
```

Installation Compilateur Java :

Maintenant on va passer à l'installation du compilateur java pour que le serveur puisse exécuter des programmes java.

```
sudo apt update  
sudo apt install openjdk-11-jdk
```

Installation Interface Graphique :

Il nous manque des paquets pour qu'on puisse avoir l'interface graphique du serveur maître, on va donc les télécharger :

```
sudo apt update  
sudo apt libxcb-xinerama0 libxcb-cursor0 libxkbcommon-x11-0
```

Problème de fermeture de Serveur :

Si les serveurs esclaves sont mal fermés, on peut les fermer avec le PID en utilisant la commande

```
sudo lsof -i -P -n  
kill -9 <PID>
```

Pour cloner le projet du github sur votre machine Linux :

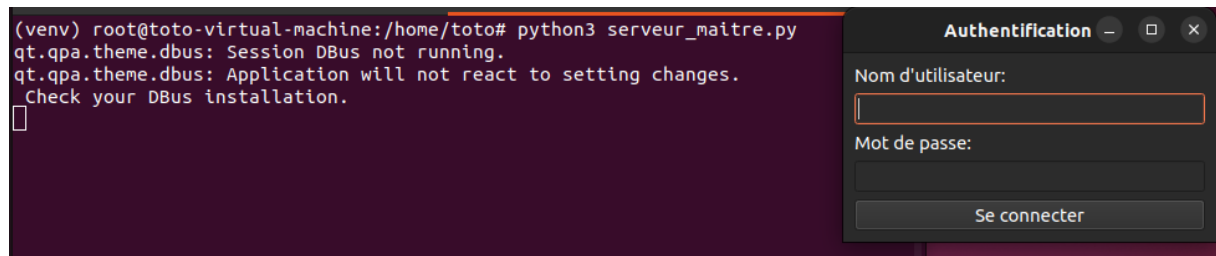
```
git clone https://github.com/Mariusklz/Cluster-de-Calculs.git
```

Lancement du projet :

Pour lancer le projet sur Linux, il faut récupérer sur mon GitHub le dossier du projet puis le lancer dans un terminal et exécuter ces commandes :

```
source venv/bin/activate  
python3 serveur_maitre.py  
python3 serveur_gcc_python.py  
python3 serveur_python,jave.py
```

Serveur maître :



Serveur Esclave 1 :

```
(venv) root@toto-virtual-machine:/home/toto# python3 serveur_gcc_python.py  
Serveur esclave démarré sur 0.0.0.0:5002  
En attente de la connexion du serveur maître...
```

Serveur Esclave 2 :

```
(venv) root@toto-virtual-machine:/home/toto# python3 serveur_python,java.py  
Serveur esclave démarré sur 0.0.0.0:5001  
En attente de connexions...
```