TP02 – CLasse, Objet, Attributs et Méthodes II

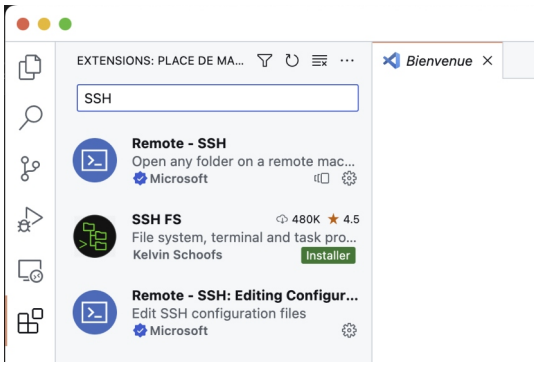
Les objectifs de ce TP sont la prise en main de l'environnement de développement et le codage des premiers programmes en langage Python orienté objet.

Partie 1 – Préparation de l'environnement de développement : VS Code + Sense Hat

Dans cette partie on prépare la configuration de VS Code pour développer des programmes en langage Python sur une cible Raspberry Pi avec un chapeau Sense Hat.
On accède à la carte Raspberry Pi via le protocole réseau SSH depuis le poste de travaux pratiques .

Étape 1 – Activation des extensions locales de VS Code

Sélectionner l'icône du gestionnaire d'extensions à partir du bandeau vertical à gauche de la copie d'écran ci-dessous.



Activer ou installer les extensions de la liste suivante :

- French Language Pack → menus et informations de l'interface en français
- Remote SSH → accès distant au système cible
- GitHub Theme → thèmes de couleurs de l'interface utilisateur

Étape 2 – Accéder au système cible Raspberry pi via SSH

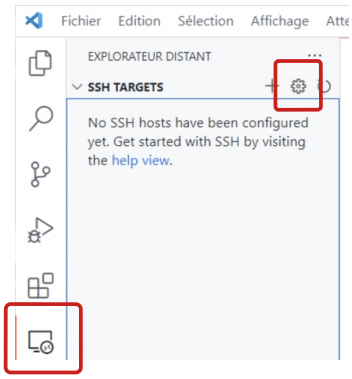
La vidéo suivante présente la configuration de l'extension Remote SSH pour accéder au système cible Raspberry pi : https://inetdoc.net/video/21E03_RPi+SenseHat+VSCode.m4v

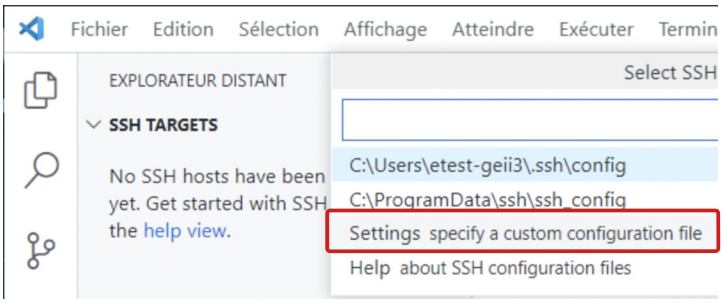

À partir de cette étape, on attribue un numéro de système cible et une adresse IPv4 à conserver pour toute la durée du cycle de travaux pratiques.
Veillez à conserver ce numéro de système et cette adresse IPv4.

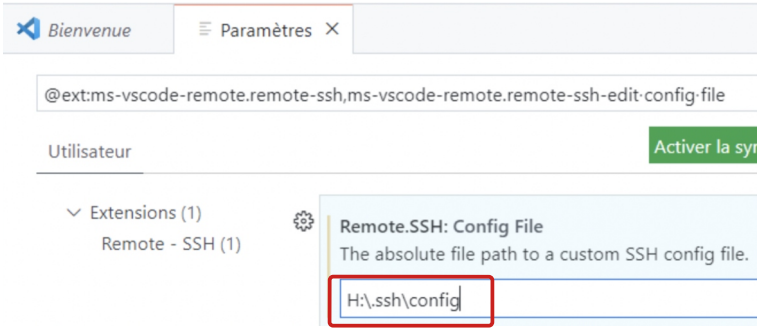
Numéro de système / Adresse IPv4 :

- ***Définition du fichier de configuration SSH***

Sélectionner la « roue » de configuration et entrer le chemin du fichier de configuration.

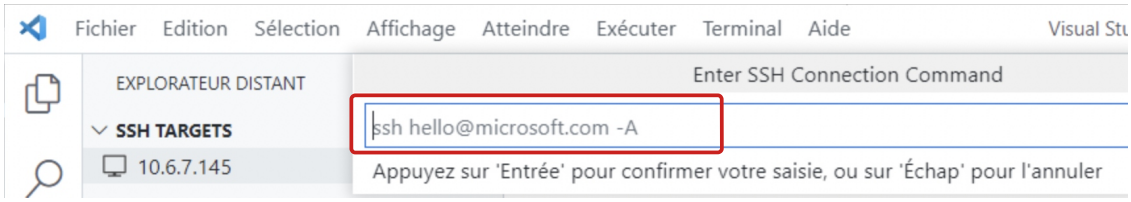




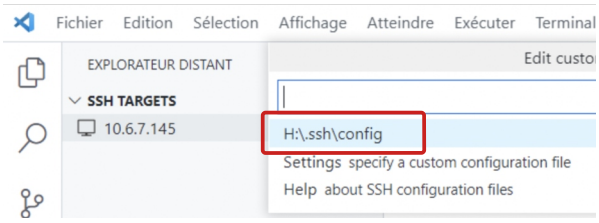


- ***Ouvrir une nouvelle connexion SSH***

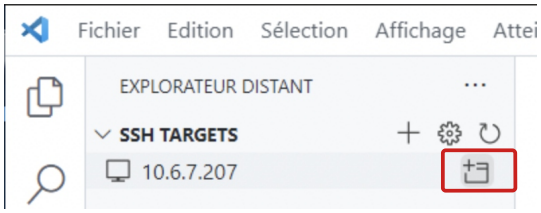
Sélectionner le caractère '+' et entrer le nom d'utilisateur et l'adresse IPv4 du système cible.
Voici un **exemple** : ssh etest-geii3@10.6.7.207 -A



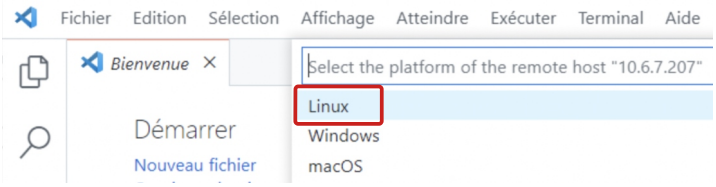
Confirmer l'enregistrement des paramètres dans le fichier de configuration.



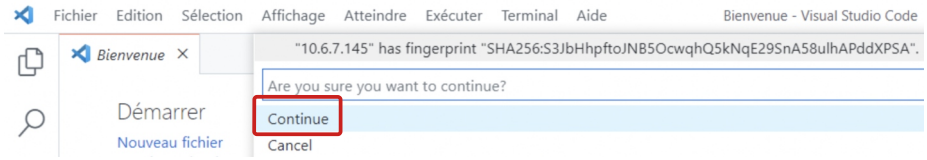
Ouvrir une nouvelle connexion.



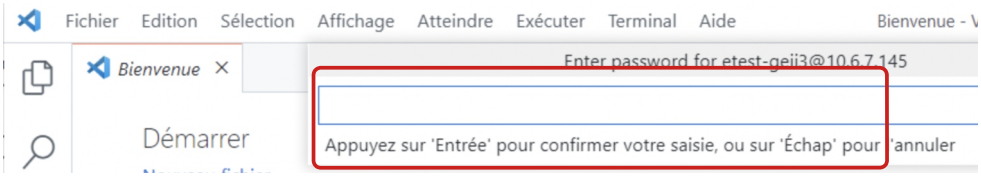
Sélectionner Linux pour désigner le système cible.



Accepter la clé publique SSH du système cible



Entrer le mot de passe le l'utilisateur



Copie d'écran partielle du résultat attendu avec l'adresse IPv4 du système cible en bas à gauche.

Étape 3 – Installer les extensions sur le système cible Raspberry pi

La vidéo suivante présente l'installation des extensions de Visual Studio Code sur le système cible Raspberry pi : https://inetdoc.net/video/21E04_RPi+SenseHat+VSCode.m4v
Pour le développement en langage Python, l'extension essentielle est :
Python Extension Pack

Étape 4 – Ouvrir un nouveau projet de programmation Python

Voici un tout premier exemple de programme de test de l'utilisation de la matrice de leds du chapeau Sense Hat.

```
#!/usr/bin/python3

import time
from sense_hat import SenseHat

sense = SenseHat()

X = [255, 0, 0]  # Red
O = [255, 255, 255]  # White

question_mark = [
O, O, O, X, X, O, O, O,
O, O, X, O, O, X, O, O,
O, O, O, O, O, X, O, O,
O, O, O, O, X, O, O, O,
O, O, O, X, O, O, O, O,
O, O, O, X, O, O, O, O,
O, O, O, O, O, O, O, O,
O, O, O, X, O, O, O, O
]

sense.set_pixels(question_mark)

time.sleep(3)

sense.clear()
```

1. Créez les dossiers POO_S5/TP1 pour stocker les fichiers source de la première séance de travaux pratiques
2. Créez un nouveau fichier nommé question_mark.py
3. Copiez le code ci-dessus dans ce fichier
4. Exécutez le code et vérifiez que la matrice de leds affiche bien un point d'intérrogation
5. Faites valider

Exercice 2 – Moniteur de température

1 – Préparation avant la séance de TP

Dans cet exercice, vous allez créer une classe TemperatureMonitor qui utilisera la bibliothèque Sense HAT pour mesurer la température et afficher une alerte si la température dépasse un seuil spécifié.

Le code de départ vous est donné ci-dessous :

```
from sense_hat import SenseHat

class TemperatureMonitor:
    def __init__(self, seuil_alerte):
        self.sense = SenseHat()
        self.seuil_alerte = seuil_alerte

    def mesurer_temperature(self):
        # Code pour mesurer la température avec la bibliothèque Sense HAT
        pass

    def afficher_alerte(self):
        # Code pour afficher une alerte sur la matrice LED
        pass

#Programme principal
# Instanciation de la classe TemperatureMonitor avec un seuil d'alerte de 25°C
temperature_monitor = ??????   (25)

# Mesure de la température et affichage d'une alerte si nécessaire
while True :
      temperature_monitor. ????? ()
      temperature_monitor. ????()
```

Représentez avec un diagramme de classes, la classe TemperatureMonitor et SenseHat (pour cette dernière ne pas spécifier les méthodes)

2 – Réalisation en séance

1) Ecrire la classe **TemperatureMonitor** avec les méthodes suivantes :
   a) **__init__(self, seuil_alerte)**: Initialise la classe en prenant un seuil d'alerte en degrés Celsius.
   b) **mesurer_temperature(self)**: Utilise la bibliothèque Sense HAT pour mesurer la température ambiante.
   c) **afficher_alerte(self)**: Affiche un message d'alerte sur la matrice LED si la température mesurée dépasse le seuil d'alerte.
2) Coder le programme principal :
   a) Instanciez la classe **TemperatureMonitor** avec un seuil d'alerte de votre choix.
   b) Utilisez les méthodes de la classe pour mesurer la température et afficher une alerte si nécessaire.
3) Testez sur https://trinket.io/sense-hat, puis sur la cible et faites valider

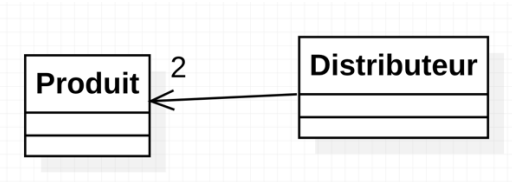
Exercice 3 – Distributeur de 2 produits

1 – Préparation avant la séance de TP

L'objectif est de créer une application du type distributeur qui gère des produits (coca, icetea, mojito). Dans un premier temps le distributeur de contiendra que 2 produits.

1) Il vous est demandé de compléter le diagramme de classes ci-dessous avec les attributs et les méthodes spécifiées :

- La classe Produit aura comme attributs un nom_produit, une quantité, un prix, et un caractère qui sera le code produit
- La classe Distributeur aura comme attributs 2 produits (produit1 et produit2), et une méthode ajouter_produits(p1, p2) qui recopie les produits p1 et p2 dans les attributs de Distributeur. Elle aura aussi une méthode afficher_produits qui affichera pour les 2 produits leurs attributs (sur la console avec print ou sur l'afficheur LCD).



2) Le programme principal est donné ci-dessous

```
if __name__ == "__main__":
    produit1 = Produit("Chips", 10, 1.5)
    produit2 = Produit("Soda", 20, 2.0)

    distributeur = Distributeur()
    distributeur.ajouter_produits(produit1,produit2)

    distributeur.afficher_produits()
```

En plus du programme principal, quelles sont les méthodes/classes qu'il faudrait modifier pour pouvoir ajouter un troisième produit ?

2 – Réalisation en séance

1) Implémenter les Classes Produit et Distributeur ainsi que le programme principal
2) Testez et faites valider

Exercice 4 – Distributeur de n produits

1 – Préparation avant la séance de TP

Nous allons maintenant modifier le programme précédent pour que le distributeur ait une liste de produits (et pas uniquement 2). Pour cela vous aurez à utiliser les listes de python.

1) Comment est déclarée une liste vide en python ?
2) Comment est ajouté un nouvel élément à la fin d'une liste python ?

2 – Réalisation en séance

1) Écrire la nouvelle classe Distributeur contenant comme attribut une liste de produits
2) Écrire la méthode ajouter_produit(self,p) à la classe Distributeur qui ajoute le produit p à la liste de produits du distributeur
3) Écrire la méthode afficher_produits à la classe distributeur qui affiche sur la console la liste des produits (cette méthode appellera la méthode afficher de la classe Produit).
4) Écrire le programme principal qui créer plusieurs produits, les ajoute au distributeur, et appelle afficher_produits
5) Tester et faites valider

Exercice 5 – Distributeur de n produits avec affichage code et joystick

1 – Préparation avant la séance de TP

L'objectif de cet exercice est d'afficher sur l'écran LCD du sense hat les codes (un caractère) des produits contenus dans le distributeur si la quantité est supérieure à 0. L'affiche de chaque produit se fera en déplaçant le joystick vers la droite ou vers la gauche.

Vous devrez aller chercher les fonctions pythons pour la manipulation du joystick sur :

https://projects.raspberrypi.org/en/projects/getting-started-with-the-sense-hat

2 – Réalisation en séance

1) Ajouter au code précédent une méthode run() à la classe Distributeur, permettant dans une boucle infinie de :
    a. Lire un événement (event) d'appui sur le joystick
    b. Si l'évènement est une pression vers la droite alors on affiche sur le LCD le code du produit suivant dans la liste
    c. Si l'évènement est une pression vers la gauche alors on affiche sur le LCD le code du produit précédent dans la liste

Pour cela vous serez être amenés à utiliser un index que vous incrémenterez ou décrémenterez et qui pourra être un attribut de la classe distributeur initialisé à 0.

Tester et faire valider

2) Extension : au lieu du code (un charactère), affichez un symbole représentant le produit.
3) Tester et faire valider