

Formation Python

Atelier Pratique AP-PY12

Atelier Pratique AP-PY12 : Exercice 12.8

Objectif :

Créer une application de *chatting* graphique comme suit :

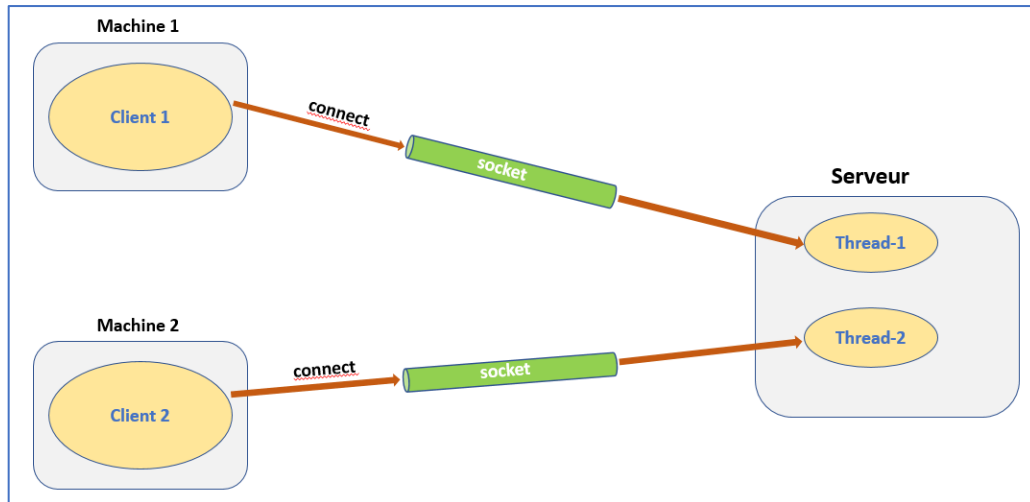
- **user 1** : interface graphique avec Tkinter
- **user2** : interface graphique avec Tkinter



Atelier Pratique AP-PY12 : Exercice 12.8

Bouton: Connect

Création d'un **thread** sur le serveur à l'écoute du client



- Si le client est déjà connecté, afficher un pop-up :
Une connexion est déjà en cours
- Si connexion avec succès, afficher un pop-up :
La connexion a été établie

Connect Disconnect

Votre message ? Bonjour à tous

Envoi

CHATTING Refresh

Server : : User 1 Vous êtes connecté au serveur. Envoyez vos messages; FIN pour quitter

Server : : User 2 Vous êtes connecté au serveur. Envoyez vos messages; FIN pour quitter

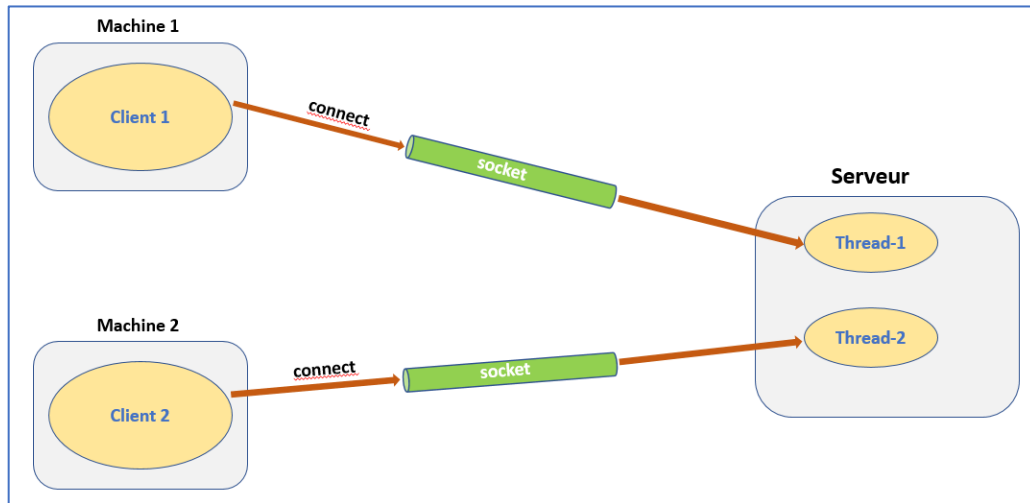
user-1 : Bonjour à tous

user-2 : Salut user 1 , comme ca va ?

Atelier Pratique AP-PY12 : Exercice 12.8

Bouton: Disconnect

Envoyer le message **FIN** au serveur, et fermer la socket du client



- Si le client est déjà déconnecté, afficher un pop-up :
Pas de connexion en cours
- Si déconnexion avec succès, afficher un pop-up :
La connexion a été fermée

The screenshot shows a web-based chat application. At the top, there is a yellow header with 'Connect' and 'Disconnect' buttons. Below the header is a text input field with the placeholder 'Votre message ?' and the text 'Bonjour à tous', followed by an 'Envoi' button. The main content area is light blue and contains a 'CHATTING' label, a 'Refresh' button, and a chat log. The chat log shows messages from the server and users:

```
Server : : User 1 Vous êtes connecté au serveur. Envoyez vos messages; FIN pour quitter
Server : : User 2 Vous êtes connecté au serveur. Envoyez vos messages; FIN pour quitter
user-1 : : Bonjour à tous
user-2 : : Salut user 1 , comme ca va ?
```

Atelier Pratique AP-PY12 : Exercice 12.8

Bouton: Envoi

Envoyer le message au serveur

- Si le client est déjà déconnecté, afficher un pop-up :
Pas de connexion en cours

The screenshot displays a web-based chat application interface. At the top, there are two buttons: "Connect" and "Disconnect". Below them is a text input field labeled "Votre message ?" containing the text "Bonjour à tous". A button labeled "Envoi" is positioned below the input field. The interface is divided into a yellow header area and a light blue main content area. In the blue area, there are two buttons: "CHATting" and "Refresh". Below the buttons is a chat log with the following messages:

- Server : : User 1 Vous êtes connecté au serveur. Envoyez vos messages; FIN pour quitter
- Server : : User 2 Vous êtes connecté au serveur. Envoyez vos messages; FIN pour quitter
- user-1 : Bonjour à tous
- user-2 : Salut user 1 , comme ca va ?

Atelier Pratique AP-PY12 : Exercice 12.8

Serveur :

1. Appeler une fonction **create_db()** qui crée une base SQLite **Echanges.db** et une table avec 2 colonnes :

```
CREATE TABLE Echanges( user TEXT, message TEXT )
```

2. Reçoit les messages des clients et les stocke dans la base de données
3. Si le client envoie **FIN** ou **RC** pour fermer la connexion avec le client
4. Si le nombre de clients dépasse la limite **NB_CLIENTS_MAX=5**,
 - . Refuser la connexion
 - . Envoyer le message **Nb maximum de clients dépassé** au client, en l'insérant dans la base de données

Atelier Pratique AP-PY12 : Exercice 12.8

Client

1. Définir une fonction **Afficher_chats(Frame)**
 - lit la base **SQLite Echanges.db**
 - affiche la liste des échanges dans la Frame bleue
2. Définir une fonction **refresh_chats()**
 - Appelle **Afficher_chats(Frame)** toutes les **2 secondes**
3. Lancer la fonction **refresh_chats()** dans un **Thread**
 - Le thread doit être démarré à la fin du connect
 - Le thread doit être arrêté après du disconnect



Atelier Pratique AP-PY12 : Exercice 12.8

Client

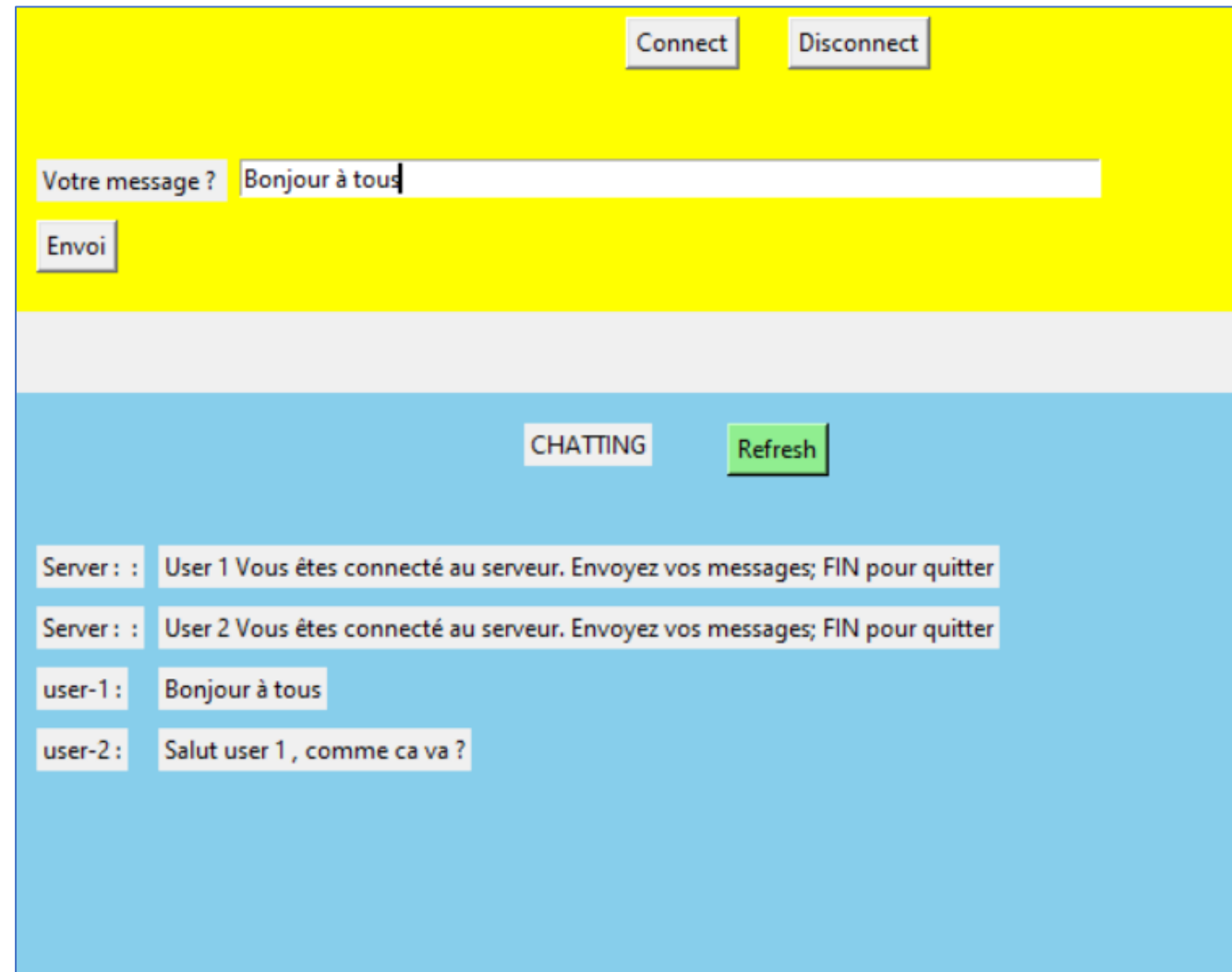
1. Définir une fonction **open_fenetre(fenetre)**

- Crée la Frame1 en jaune
- Crée la Frame2 en bleue

2. Lancer **2 clients graphiques en parallèle**

Pour cela, il faut utiliser la librairie **multiprocessing**

Voir exemple dans la page suivante



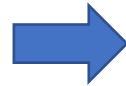
Atelier Pratique AP-PY12 : Exercice 12.8

Multiprocessing

```
import multiprocessing

def program1():
    window1 = tk.Tk()
    open_fenetre( window1 )
    window1.mainloop()

def program2():
    window2 = tk.Tk()
    open_fenetre( window2 )
    window2.mainloop()
```



```
if __name__ == "__main__":

    # Création des processus
    process1 = multiprocessing.Process(target=program1)
    process2 = multiprocessing.Process(target=program2)

    # Lancement des processus
    process1.start()
    process2.start()

    # Attente de la fin des processus
    process1.join()
    process2.join()
```

Atelier Pratique AP-PY12 : Exercice 12.8

