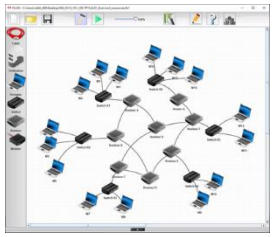
1. Logiciel Filius



TP SNT proposé par un collègue à des élèves de seconde en S.N.T.
Fichier : 1_NSI TP15.zip à récupérer sur le site www.didrit.fr

2. Commandes bash

Tester les commandes ipconfig, ping et tracert sur votre ordinateur avec le réseau local et avec le réseau Internet.

3. Communiquer entre deux machines en Python

Attention : Cet exercice est à faire en binôme !
Voici deux programmes nommés « serveur.py » et « client.py » à recopier sur deux machines différentes du même réseau.

Programme : client.py

```
import socket as sk

hote = "127.0.0.1"
port = 12800
serveur = sk.socket()
serveur.connect((hote, port))
print("Connexion établie avec le serveur", hote, "sur le port", port)
msg_a_envoyer = b""
while msg_a_envoyer != b"fin" :
    msg_a_envoyer = input("> ")
    msg_a_envoyer = msg_a_envoyer.encode()
    serveur.send(msg_a_envoyer)
    msg_recu = serveur.recv(1024)
    print(msg_recu.decode())
print("Fermeture de la connexion")
serveur.close()
```

Programme : serveut.py

```
import socket as sk

hote = "127.0.0.1"
port = 12800
connexion = sk.socket()
connexion.bind((hote, port))
connexion.listen(1)
print("Le serveur écoute sur le port", port)
client, infos = connexion.accept()
print("Connexion du client", infos)
msg_recu = b""
while msg_recu != b"fin" :
    msg_recu = client.recv(1024)
    print("Reçu :", msg_recu.decode())
    client.send(b"Ok")
print("Fermeture de la connexion")
client.close()
connexion.close()
```

a) A quoi correspond l'adresse '127.0.0.1' présente dans les deux programmes ? Par quelle adresse faut-il la remplacer ?
b) Ajouter un commentaire à la fin de chaque ligne pour expliquer leur rôle.
c) Tester ces deux programmes

Remarque : Les transferts de messages ne se font pas avec des chaines de caractères au format *str*, mais au format *bytes*, il faut les convertir : b"Ok" équivaut à *bytes*("Ok", "UTF-8")

On a réussi à communiquer entre deux machines, mais impossible avec ces programmes de communiquer avec une troisième machine… le serveur n'écoute ici que le premier client qui s'est connecté !

4. Communiquer entre plus de deux machines en Python

Attention : Cet exercice est à faire en trinôme (ou plus) !

Voici une nouvelle version du programme serveur nommée « serveur_plus.py », elle fonctionne avec le même programme « client.py » de l'exercice précédent mais accepte maintenant plusieurs clients !

a) Penser à modifier l'adresse de l'hôte !
b) Quelle est la différence entre les messages "fin" et "stop" ?
c) Ajouter un commentaire à la fin de chaque ligne pour expliquer leur rôle.
d) Tester cette version avec plusieurs machines simultanément.

Remarque : Pour réaliser une vraie messagerie instantanée, il faut utiliser de la programmation événementielle et non pas de la programmation séquentielle, ou gérer les processus en faisant du multitâche. En effet le programme client se met en pause lorsqu'il attend que l'utilisateur entre son message (*input*) puis se met en pause lorsqu'il attend la réponse du serveur (*client.recv*).

Plus de détails sur le site OpenClassRooms.com :
https://openclassrooms.com/fr/courses/235344-apprenez-a-programmer-en-python/234698-gerez-les-reseaux

ou sur le site Développez.com
https://python.developpez.com/cours/apprendre-python3/?page=page_20

```
import socket as sk
import select as slc

hote = "127.0.0.1"
port = 12800
connexion = sk.socket()
connexion.bind((hote, port))
connexion.listen(5)
print("Le serveur écoute sur le port :", port)
serveur_actif = True
liste_clients = []
while serveur_actif :
    liste_demandes, wlist, xlist = slc.select([connexion], [], [], 0.05)
    for demande in liste_demandes :
        client, infos = demande.accept()
        print("Connexion du client", infos)
        liste_clients.append(client)
    liste_a_lire = []
    try :
        liste_a_lire, wlist, xlist = slc.select(liste_clients, [], [], 0.05)
    except slc.error :
        pass
    else :
        for client in liste_a_lire :
            msg_recu = client.recv(1024)
            infos = client.getpeername()
            print(infos, ":", msg_recu.decode())
            client.send(b"Ok")
            if msg_recu.decode() == "fin" :
                print(infos, "est parti !")
                liste_clients.remove(client)
                client.close()
            elif msg_recu.decode() == "stop" :
                serveur_actif = False
print("Fermeture des connexions")
for client in liste_clients :
    client.close()
connexion.close()
```