16 DECEMBRE 2022

SAE 3.02

Develop communicating applications

DOCUMENTATION

Development part

JULES BRUTSCHY
RT211
Cyber - FI

Table des matières

1. Introduction 2
   a. Use of the application 2
2. Application structure 2
3. Code architecture 3
   a. Client.py 3
   b. Commandes.py 3
   c. GUI.py 3
      I. App 3
      II. NvServ 4
      III. Using your own file as server list 4
   d. Mainserveur.py 4
4. What has been done 4
5. What doesn't work 4

1. Introduction

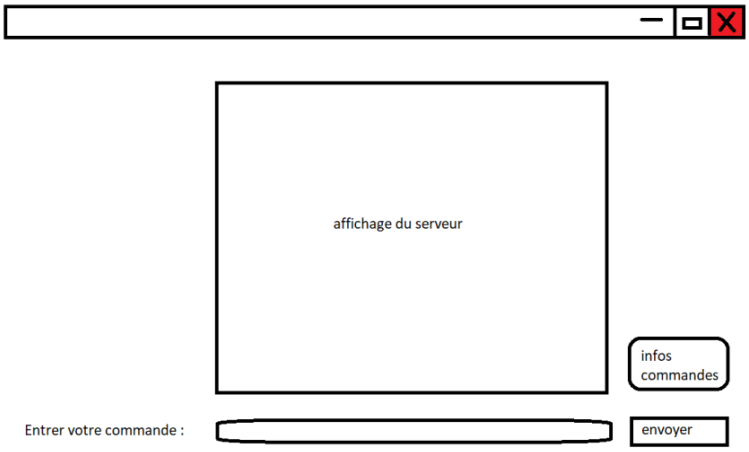
a. Use of the application

This application enables us to monitor servers or clients' machines trough a simple graphical interface.

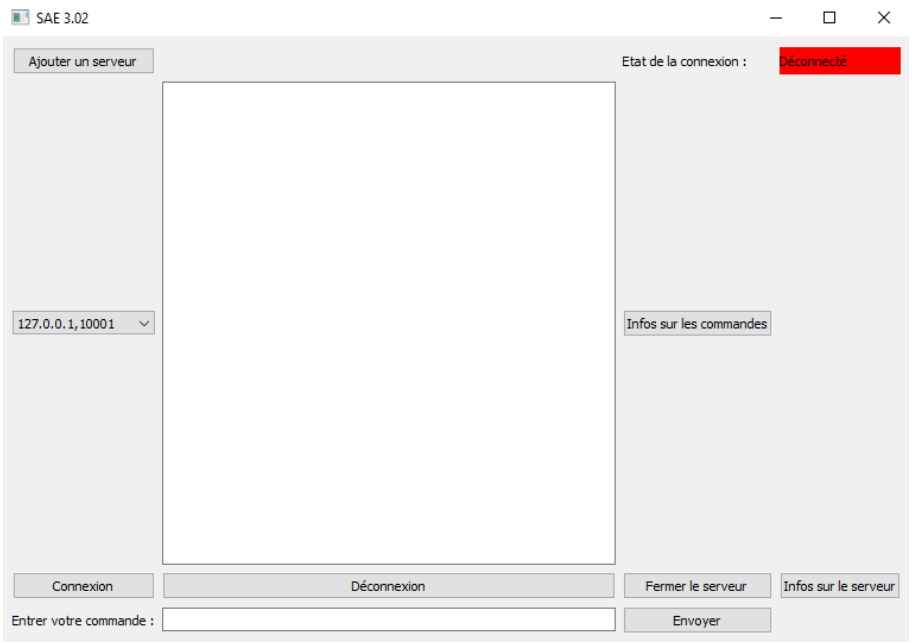
2. Application structure

The structure of the application is as follows, a principal widget in which dialog boxes can be opened.

The schema on which I based myself to build the principal widget is the following.



Several buttons were added.



3. Code architecture

a. Client.py

This file allows you to create a client and use it in a client-server communication, it contains a class called “Client” in which I defined all the main tools for the communications.

The names of the functions are explicit and reflect their functionality.

I decided to put the connection I a socket and to thread it, in order to be able to connect the client to multiple servers and being able to send messages independently of the reception of the messages from a server.

b. Commandes.py

This file contains all the commands that are necessary for the operation of the application. Once again, the names of the functions are explicit and reflect their functionality.

For the command using the OS of the server, I used “subprocess” which is a module allows you to spawn new processes, connect to their input/output/error pipes, and obtain their return codes.

For all the others commands I used psutil and platform.

For the graphic of usage of the CPU I used matplotlib.

c. GUI.py

This file contains the graphical interface, it has been divided in 3 classes:

- App: the main window of the GUI, for the connection and sending/receiving messages to/from the server.
- NvServ: the window that allows us to define the server address and the port.
- Infos: the window that displays graphical informations.

I. App

Explanation of the different functions:

_actionInfo allows the opening of a window which gives information about the commands.
_actionCo allows the connection.
_actionDeco allows disconnection.
_actionFermSrv allows you to configure the server shutdown.
_actionEnv allows you to configure the sending of a message.
_actionNvserv triggers the class “NvServ”.
_actionInfos triggers the class “Infos”.
_closeEvent allows to close the application from the Windows cross.

II. NvServ

The management of the different addresses and ports is done in a .txt file (host.txt) which contains the information in this form:

```
127.0.0.1,10001
127.0.0.1,10002
127.0.0.1,10000
```

The differents servers are separated by a line break.

```
letxt = open("host.txt", "r")
text = letxt.read()
text = text.split("\n")
```

```
host = self.__list.currentText().split(",")[0]
port = int(self.__list.currentText().split(",")[1])
```

The difference betwin the address and the port is done by splitting the line at the comma level

III. Using your own file as server list

If you want to use your own file, you will just need to replace the name in the code of your own file, or rename your file in “host.txt”.

The structure of your file must be such that : differents server are separated by a line break and the the address and the port are separated by a comma.

d. Mainserveur.py

This file allows you to create a server and use it for client-server communication, it contains a class called “Serveur” in which I defined all the main tools for the communications.

4. What has been done

Except as noted in the next section, all requested features are functional.
Furthermore, the CPU usage graph is displayed when the CPU command is entered.

5. What doesn't work

When using the disconnect command, the client is disconnected from the server, but you cannot connect it on the same port again, you must close the server and retry.

When using the reset command, the server close but doesn’t open again.