Projet dev réseau

Backlog

Robin Penea

Objectif

Mettre en pratique les sockets dans un projet bien architecturé

Robin Penea

☘ Libertés ☘

- Thème entièrement libre
  - Whiteboard en réseau
  - Partage de fichiers (Dropbox like)
  - Streaming vidéo / audio
  - Screen share
  - Chat audio
  - Monitoring de ressources systèmes
  - Jeu multi
- `flet` non obligatoire, vous utilisez ce que vous voulez

💁‍♀️ Conseils généraux 💁‍♀️

Protocole

- Démarrez rapidement des échanges "de base" entre client et serveur
- Après ça, prenez le temps de poser le protocole sur papier / tableau de façon **exhaustive** :
  - Liste de **tous** les messages
  - Grandes séquences d'actions
- Toute l'équipe doit avoir une vision claire du protocole

💁‍♀️ Conseils généraux 💁‍♀️

Flet

- Ne faites pas de navigation de pages, restez sur une seule page
- Utilisez plutôt des Dialogs (Pseudo + connexion, choix de room, etc.)
- Ce n'est pas une limite de Flet, mais ça implique une gestion de state plus complexe, avec des callbacks réseau, etc
- **Ca reste un cours réseau, pas un cours de Flet**
  - Personne ne doit passer 3 jours à faire de l'UI, sinon le module n'est pas validé

⛓️ Contraintes ⛓️

- Utiliser les sockets en Python (TCP et / ou UDP)
- Toujours N clients et 1 serveur
- Les erreurs doivent être gérées
  - Message d'erreur affiché, mais pas de crash
  - **Soyez laxiste en réception, strict en émission**
- Tests unitaires attendus au moins pour le protocole

⛓️ Contraintes ⛓️

- Le protocole doit être **custom** (JSON ou équivalent interdit)
  - Vous pouvez utilisez des JSON dans le payload, mais pas dans le protocole
- L'objectif est de proposer une stack **réutilisable** dans d'autres projets
  - On aurait en plug'n'play un système de room et de discussion dans un projet qui n'a rien à voir le votre !

Story #01

EN TANT QUE Développeur

JE VEUX Créer un fichier `PROTOCOL.md`

AFIN DE Décrire mon protocole de façon *exhaustive*

DoD Fichier `PROTOCOL.md` à la racine du projet

💁‍♀️ Info 💁‍♀️

- ⚠️ **Tous les messages ne doivent pas être codés *dans* le protocole**,
  N'oubliez pas :
  - Définissez bien les frontières entre le protocole et l'application
  - Le protocole doit se limiter à faciliter certaines séquences d'actions
  - Pour vous aider, *pensez que votre protocole doit être utilisable par une autre application*
- Vous trouverez un exemple de `PROTOCOL.md` dans le projet d'exemple

💁‍♀️ Info 💁‍♀️

- Décrivez tous les messages et réponses attendus
  - Pour chaque message, le format + exemple
- Si une séquence > 2 messages, faites un diagramme de séquence
  - Utilisez mermaidjs https://mermaid.js.org/ (promis c'est pépite)
  - Intégrable directement dans le .md et preview sur GitHub
  - Plugin IntelliJ https://plugins.jetbrains.com/plugin/20146-mermaid
  - Plugin VS Code https://marketplace.visualstudio.com/items?itemName=bierner.markdown-mermaid

Story #02

EN TANT QUE Client

JE VEUX Choisir un pseudo à la connexion

AFIN DE Personnaliser l'expérience utilisateur

DoD Chaque client est identifié par un pseudo

💁 Info 💁

- Tant que le pseudo n'est pas choisi, le client n'est pas connecté
- Sa connexion est notifié aux autres *après* le choix du pseudo

Story #03

EN TANT QUE Client

JE VEUX Rejoindre une room après le pseudo

AFIN DE Communiquer avec un groupe restreint

DoD Les messages sont échangés avec la room choisie

💁‍♀️ Info 💁‍♀️

- Les rooms sont déjà créées en dur par le serveur (pour vous alléger le travail)
- Le client peut voir la liste des rooms après son choix de pseudo
- Le client peut rejoindre une room existante

Story #04

EN TANT QUE Serveur

JE VEUX Maintenir plusieurs sockets clients

AFIN DE Gérer autant de clients que possible

DoD 2+ clients sont connectés en simultané

Story #05

EN TANT QUE Client

JE VEUX envoyer des données à ma room

AFIN DE Echanger sur le réseau !

DoD Les autres clients reçoivent le message envoyé

💁 Info 💁

- Le message peut être du texte et / ou du binaire (en fonction de votre sujet)

Story #06

EN TANT QUE Client

JE VEUX Envoyer un message déclenchant une séquence

AFIN DE Supporter des actions complexes dans le protocole

DoD L'envoi d'un message provoque un état intermédiaire

💁 Info 💁

- Certains messages déclenchent une machine à état intermédiaire
- Tant que l'état intermédiaire n'est pas résolu, l'émetteur est bloqué
- Exemples
  - Action de jeu demandant aux autres un vote
  - Envoi d'un fichier avec attente de confirmation
  - Demande de démarrage de streaming audio

Story #07

EN TANT QUE Client / Serveur

JE VEUX Spécifier la taille des données envoyées

AFIN DE Faciliter la lecture socket des destinataires

DoD Chaque message contient la taille des données envoyées

💁‍♀️ Info 💁‍♀️

- Stocké sur un int (32 bits)
- Présent dans tous les messages
- Cette taille concernera les données envoyées *après* le message

Story #08

EN TANT QUE Serveur

JE VEUX avoir une notification de connexion à une room

AFIN DE Informer les clients déjà connecté

DoD Tous les clients connectés (sauf le nouveau) reçoivent un message d'info `"PSEUDO s'est connecté"`

Story #09

EN TANT QUE Admin

JE VEUX Un dashboard admin

AFIN DE Administrer facilement le serveur

DoD Une UI (flet ou autre) avec tous les clients connectés

💁‍♀️ Info 💁‍♀️

- **Ce n'est pas un utilisateur qui a des droits spéciaux lorsqu'il se connecte**
- L'interface d'admin doit tourner sur la machine du serveur. Ca tourne dans le même programme que `server_socket`
- Interface ultra simplifiée pour qu'on voit tout en un coup d'oeil
  - Affiche en temps réels les clients connectés
  - Adresse IP + port
  - Pseudo
  - Room
  - Date du dernier message envoyé

Story #10

EN TANT QUE Admin

JE VEUX Serveur peut kicker les users

AFIN DE Maintenir les trolls sous contrôle

DoD Un bouton dans l'UI pour déconnecter un client

💁‍♀️ Info 💁‍♀️

- Dialog de confirmation
- L'utilisateur est déconnecté
- Message aux autres clients `"PSEUDO a été kické"`

Story #11

EN TANT QUE Serveur

JE VEUX Envoyer un message en broadcast

AFIN DE Prévenir d'évènements importants

DoD Un message peut être envoyé à all / room / MP

💁‍♀️ Info 💁‍♀️

- Un code spécial qui affiche une Dialog / Notification avec le message du serveur
- Ex : `"Message du server le 31/01/2024 10h42 : ...."`

Story #12

EN TANT QUE Serveur

JE VEUX Etablir une connexion P2P entre 2 clients

AFIN DE Permettre une communication privée

DoD 2 clients s'échangent des messages sans passer par le serveur

💁‍♀️ Info 💁‍♀️

- C'est le serveur qui initie la connexion entre les 2 clients
  - Il connait l'adresse IP de A et B
  - Il communique à chacun l'adresse IP de l'autre
- A partir de là, A et B seront client *et* serveur entre eux
- Messages envoyés sur une room peuvent l'être en P2P

Story #B01

EN TANT QUE Serveur

JE VEUX Implémenter un load balancer basique

AFIN DE Répartir la charge entre plusieurs instances serveur

DoD Un serveur maître peut rediriger les clients vers des serveurs secondaires

💁‍♀️ Info 💁‍♀️

- Serveur maître maintient la liste des serveurs actifs
- Si plus de 3 (variable) clients connectés, le serveur maître redirige les clients vers des serveurs secondaires

Story #B02

EN TANT QUE Client

JE VEUX Avoir un système de heartbeat avec le serveur

AFIN DE Détecter rapidement les déconnexions inattendues

DoD Un message PING/PONG est échangé toutes les 30 secondes

💁 Info 💁

- Le serveur envoie périodiquement un PING à chaque client
- Le client doit répondre avec un PONG dans les 5 secondes
- Si pas de réponse, le client est considéré comme déconnecté
- Permet de nettoyer les connexions fantômes (client crashé, perte réseau)
- Utilise des timers pour la gestion asynchrone

Story #B03

EN TANT QUE Développeur

JE VEUX Ajouter une couche de compression des messages

AFIN DE Optimiser la bande passante utilisée

DoD Les messages texte sont compressés automatiquement avant envoi

💁 Info 💁

- Utilisation de zlib ou gzip pour la compression
- Header dans le protocole indiquant si le payload est compressé
- Compression activée uniquement si gain > 10% (évite overhead inutile)
- Décompression transparente côté récepteur
- Particulièrement utile pour les messages JSON ou texte long