Instituto Superior de Engenharia de Lisboa

Licenciatura em Engenharia Informática e de Computadores

Programação Concorrente

Verão de 2022/2023, Terceira Série de Exercícios

---

1. Realize a classe **`MessageQueue<T>`** para comunicação de mensagens entre *coroutines*, através de filas de mensagens *first in first out*, disponibilizando os métodos:
    - **`suspend fun enqueue(message:T): Unit`**
    - **`suspend fun dequeue(timeout: Duration): T`**

    Ambos os métodos devem suportar cancelamento. A construção das filas deve suportar a definição da sua dimensão máxima.

2. Realize funções de extensão sobre as classes **`AsynchronousServerSocketChannel`** e **`AsynchronousSocketChannel`** para a aceitação de ligações, escrita e leitura, sem bloquear as *threads* invocantes. Estas funções devem ser **`suspend`**, suportar cancelamento e expor uma interface síncrona.

3. Realize um sistema servidor com interface TCP/IP para troca de mensagens entre sistemas clientes, com a funcionalidade apresentada em seguida.
    - Os sistemas clientes interagem com o sistema servidor através do envio de linhas de texto, que podem ser comandos ou mensagens. Um comando começa por '/', seguido do nome do comando e de zero ou mais argumentos. Uma mensagem é qualquer linha que não comece por '/'.
    - O sistema servidor está organizado em salas. Cada sistema cliente pode estar em zero ou em uma sala. Após ligação, os sistemas cliente não estão em nenhuma sala. Existem comandos para entrar numa sala, sair de uma sala e terminar a ligação.
    - Quando um sistema cliente está numa sala: a) pode enviar mensagens para essa sala; b) recebe todas as mensagens enviadas por outros clientes presentes nessa sala.

Os comandos que podem ser enviados por um cliente através duma ligação TCP/IP são:

- **`/enter <room-name>`** - entra na sala <room-name>, criando-a se necessário.
- **`/leave`** - abandona a sala onde se encontra.
- **`/exit`** - termina a ligação ao servidor.

O sistema deve também aceitar os seguintes comandos enviados localmente através do *standard input*:

- **`/shutdown timeout`** - inicia o processo de encerramento de servidor, deixando de aceitar ligações mas esperando que todas as ligações terminem. Todos os clientes devem receber uma mensagem notificando que o processo de encerramento teve início. Caso ainda existam clientes ligados após **`timeout`** segundos, o servidor deve terminar abruptamente.
- **`/exit`** - termina o servidor de forma abrupta.

O sistema desenvolvido deve utilizar um número de *threads* adequado à capacidade computacional da máquina hospedeira, não devendo ser proporcional ao número de clientes ligados.

Para testes manuais do servidor, use um qualquer cliente TCP/IP, tal como o telnet, o putty ou o netcat. É valorizada a inclusão de testes automáticos no projeto.

A entrega deve ser feita através da criação da tag **`1.0.0`** no repositório individual de cada aluno, ou no repositório de grupo criado para o efeito. A entrega deve também incluir um ficheiro **`README.md`** com os aspectos importantes de desenho e implementação.

O repositório https://github.com/isel-leic-pc/s2223v-se3 tem uma implementação de um sistema equivalente, usando duas *threads* por cliente ligado. Considere a organização descrita neste repositório como base para a organização do sistema a realizar.

Data limite de entrega: 11 de junho de 2023

Esta série de exercícios pode ser realizada em grupos de até três alunos.

ISEL, 15 de maio de 2023