Instituto Superior de Engenharia de Lisboa

Licenciatura em Engenharia Informática e de Computadores

Programação Concorrente, Inverno de 2017/2018

Terceira Série de Exercícios

Resolva os seguintes exercícios e apresente os programas de teste com os quais validou a correção das suas implementações.

1. Pretende-se neste exercício realizar um servidor com interface TCP para acesso a uma *key-value store*, inspirada no servidor REDIS (https://try.redis.io), suportando os seguintes comandos
   - **`SET <key> <value>`** - adiciona ou afecta a chave **`key`** (*string*) com **`value`** (*string*).
   - **`GET <key>`** - retorna o valor associado à chave **`key`**. Caso não exista, retorna **`"(nil)"`**.
   - **`KEYS`** - retorna todas as chaves existentes.
   - **`SHUTDOWN`** - termina o servidor, esperando pelo fim de todos os comandos em curso.
   - **(Opcional)** **`BGET <key> <timeout>`** - retorna o valor associado à chave **`key`**. Caso não exista, o comando fica bloqueado até que exista valor para a chave, no máximo de **`timeout`** milésimos de segundo. Retorna **`"(nil)"`** caso expire o tempo limite sem que o valor existas.

   Cada comando é definido através de uma linha de texto. A resposta pode ocupar várias linhas, terminando sempre com uma linha vazia.

   Em anexo está presente uma implementação parcial, não optimizada e síncrona para este problema. Inspirando-se na estrutura do servidor fornecido, implemente uma versão com os seguintes requisitos:

   - Utilização da variante assíncrona da operação da API de *sockets* **`TcpListener.AcceptTcpClient`** de acordo com o estilo *Asynchronous Programming Model* (APM) da plataforma .NET. (Para simplificar, não é obrigatório utilizar a interface assíncrona na leitura e na escrita dos *streams* associados ao *socket* que suporta a comunicação cliente/servidor.)
   - Limitação, por concepção, do número máximo de pedidos que o servidor pode processar simultaneamente.
   - Limitação, por concepção, do número máximo de chamadas recursivas ao método de *callback* por cada IOCP *thread*. As chamadas recursivas podem ocorrer quando a operação invocada assincronamente é concluída pelo método `BeginXxx`. Nota: a invocação recursiva do método de *callback* é reportada através da propriedade **`IAsyncResult.CompletedSynchronously`**.
   - Funcionalidade de registo suportada por uma *thread* de baixa prioridade (*logger thread*) criada para o efeito. As mensagens com os relatórios devem ser passadas das *threads* que servem pedidos (produtoras) para a *logger thread* usando um mecanismo de comunicação que minimize o tempo de bloqueio das *threads* produtoras. A funcionalidade de registo deve ter o mínimo de influência no tempo de serviço, admitindo-se inclusivamente a possibilidade de ignorar relatórios.
   - **(Opcional)** Minimização dos número de *threads* mobilizadas no servidor, nomeadamente evitando-se o bloqueio de *threads* no comando **`GET`**.

2. Implemente uma versão do servidor especificado no exercício 1 mas usando o estilo *Task based Asynchronous Pattern* (TAP) nas operações assíncronas. Tire partido dos métodos assíncronos disponíveis a partir do C# 5.0.

(Ao contrário do exposto no exercício 1, todas as operações de I/O sobre *sockets* devem utilizar a interface assíncrona ao estilo TAP que está disponível no .NET *Framework*.)

3. Usando a TPL (*Task Parallel Library*) e, se achar conveniente, os métodos assíncronos do C#, implemente um método utilitário para procurar os ficheiros de maior dimensão presentes numa dada pasta e respectivas subpastas. A aplicação recebe o caminho absoluto da pasta raiz da pesquisa e o número de ficheiros a mostrar. A pesquisa é realizada assincronamente, tirando partido da eventual existência de vários processadores e é passível de ser cancelada. O resultado da pesquisa contém:
    - A lista dos nomes dos ficheiros (i.e. caminhos absolutos) com maior dimensão;
    - O número total de ficheiros encontrados.

4. Implemente uma versão minimalista de aplicação para a realização de pesquisas no sistema de ficheiros. A aplicação contém interface gráfica para recolha dos parâmetros de pesquisa e para apresentação dos resultados. Para melhorar a experiência de utilização, a apresentação dos resultados é realizada de forma incremental enquanto a pesquisa decorre, apresentado-se os ficheiros de maior dimensão encontrados até ao momento. A interface gráfica inclui botão para cancelamento da pesquisa em curso. Faça uso da TPL e do método utilitário da alínea anterior, modificando-o se necessário.

Data limite de entrega: 3 de Janeiro de 2018

ISEL, 3 de Dezembro de 2017