Redes de Computadores — 2017/2018

Trabalho Prático de Avaliação nº 1

Instruções para testes finais e entrega do trabalho

Para o teste do trabalho deverá:

1. IGNORAR ESTE PASSO: Construir um arquivo executável **trab1.jar** com a implementação do cliente desenvolvido (ver as instruções em anexo).

2. IGNORAR ESTE PASSO: Testar o trabalho usando o script **t1_test.sh** (ver as instruções em anexo) com **trab1.jar** em parâmetro.

3. **Provisoriamente** teste o seu trabalho na diretoria que contém a diretoria **ftp17** da seguinte forma:

   Lance o servidor assim:

   java -Xbootclasspath/p:./ftp17/socket.jar ftp17/Ftp17Server

   Lance o seu cliente assim:

   java -Xbootclasspath/p:./ftp17/socket.jar ftp17/Ftp17Client file11000 localhost

Quando achar que os seus testes e a sua solução são os que vai entregar, prepare as versões finais dos ficheiros **trab1.jar** e do código do seu cliente (ex. **Ftp17Client.java** e, eventualmente outros ficheiros), e

1. Responda online ao questionário (pode emendar posteriormente a sua resposta):

   https://goo.gl/forms/KrNRo2lsVuYD4lkt1

2. Entregue o trabalho através da página:

   https://goo.gl/forms/oYD8VNZqDrPJjqaZ2

   http://polyform.di.fct.unl.pt/rc1718-tp1/index.html (tentar em alternativa)

   Siga as instruções até obter a mensagem de sucesso da submissão. Se tiver de fazer mais do que uma submissão, só a última será considerada.

Como construir um jar executável correspondente ao cliente que desenvolveu

Para testar uma versão estável do seu **Ftp17Client.java** construa um arquivo jar executável com o código do seu cliente. Notar que é apenas necessário incluir no **jar** executável o cliente e as classes de que este necessita. Para o efeito proceda da seguinte forma:

Usando o ambiente Eclipse

Siga as instruções na documentação Eclipse que pode encontrar em:
https://eclipse.org/documentation/

Procure na documentação da sua versão de Eclipse:
Creating JAR Files > Creating a New Runnable Jar

Depois de criar o jar executável pode testar se está correto e funciona, executando-o (como se fosse uma aplicação), verificando que exibe o mesmo comportamento do código do cliente que desenvolveu e que antes tinha testado.

Usando a ferramenta jar numa linha de comando (shell)

Compile o cliente (com javac) obtendo todas as classes correspondentes à implementação e execução do cliente que desenvolveu. Criar na diretoria que contém a diretoria **ftp17** um ficheiro txt (ex., **"manifest.txt"**) com uma única linha

Main-Class: classname

em que **classname** será a classe com o Main do seu cliente (ex. **ftp17.Ftp17Client**)

Para criar um jar executável chamado **trab1.jar** (que será um único ficheiro ou arquivo com extensão **.jar**) basta correr o seguinte comando jar na diretoria onde está ftp17:

jar cvmf ftp17/manifest.txt trab1.jar ftp17/*.class

Para verificar em qualquer momento o conteúdo do arquivo jar executável que foi criado dê o comando:

jar tvf trab1.jar

Deverá obter um output da forma:

```
   0        Thu Oct 13 11:37:12 WEST 2016 META-INF/
  96        Thu Oct 13 11:37:12 WEST 2016 META-INF/MANIFEST.MF
5526        Thu Oct 13 11:29:12 WEST 2016 ftp17/Ftp17Client.class
3817        Thu Oct 13 11:29:10 WEST 2016 ftp17/Ftp17Packet.class
… etc. ……
```

Para testar que pode executar o seu jar executável **trab1.jar** dê o comando:

java -jar trab1.jar

Deve observar o mesmo comportamento da execução do seu cliente que colocou no arquivo executável

Testar o cliente com o jar criado anteriormente

No arquivo de instruções para entrega do trabalho 1 encontra o seguinte ficheiro:

t1_test.sh

que é um shell script que vai usar para correr e testar o seu trabalho em vários cenários diferentes. Este shell script só funciona em Linux ou Mac OS X e requer que o computador tenha o sistema Docker (http://www.docker.com) instalado, o que já foi feito nos laboratórios nas imagens Linux.

Poderá ter que tornar o script executável com o comando:

chmod +x t1_test.sh

Para cada cenário executar o script de teste na seguinte forma:

./test_sh cenario cliente

cenario — pode ser 1, 2, 3, …...
cliente — pode ser um de dois parâmetros:
SW — para correr e testar o comportamento de um cliente Stop&Wait
trab1.jar — para correr e testar o seu cliente.jar

Os testes 1 a 3 são obrigatórios e são assim executados:

./t1_test.sh 1 SW // Obterá os indicadores de execução de um cliente S&W no cenário 1
./t1_test.sh 1 trab1.jar // Obterá os indicadores de execução do seu cliente no cenário 1

onde figura 1 podem estar 2, 3, ...

Os testes dos outros cenários são optativos e executam-se da mesma forma:

./t1_test.sh 4 trab1.jar // Obterá os indicadores de execução do seu cliente no cenário 3

onde figura 4 podem estar 4, 5, …

Nestes cenários não faz sentido testar o cliente S&W.

Registe os resultados para os inserir no form online de submissão.

Não esquecer

1) Não deve utilizar o script **t1_test.sh** em pastas (ou diretorias) com espaços ou caracteres especiais e não altere o conteúdo do script

2) Faça **chmod +x t1_test.sh** para tornar o script executável

3) É normal o script demorar algum tempo a carregar a imagem docker da primeira vez que é executado

4) No final do teste, o servidor ainda demora cerca de 15 segundos a terminar. A transferência é bem sucedida quando uma mensagem a indicar isso é mostrada.

5) Pode executar os testes tantas vezes quantas quiser.