



Mòdul 9  
DAM 2

Autor  
Unitat formativa  
Pràctica

Programació de serveis i  
processos

Helena Madrenys Planas  
UF1b – EXAMEN  
Examen – 14/03/2022

*Examen UF1b*

UF1 b: Sòcols i serveis



## Índex

<b>Exercici 1.....</b>	<b>3</b>
Servidor.....	3
Client.....	5
Output.....	6
<b>Exercici 2.....</b>	<b>7</b>
Servidor.....	7
Client.....	8
Output.....	9

## Exercici 1

### Servidor

El nom de la classe és **Ex1Servidor**. Veiem doncs el «package» creat, els imports que hem fet i el tag «@author».

En la funció main hi hem fet el «throw IOExceptions» per a estalviar-nos el «try-catch» enmig del codi. Aquesta excepció es tractarà així en tot l'examen.

```
1 package Madrenys_Helena_UF1_examen;
2
3 // @author: Helena Madrenys Planas
4
5 import java.io.DataInputStream;
6 import java.io.DataOutputStream;
7 import java.io.IOException;
8 import java.net.ServerSocket;
9 import java.net.Socket;
10
11 public class Ex1Servidor {
12     public static void main( String args[] )throws IOException{
```

Primer de tot doncs caldrà establir la connexió entre el servidor i el client. Per fer-ho, establim el port donat per l'exercici (2222), creem un «ServerSocket» per a acceptar peticions de clients i un «Socket» per a poder comunicar-nos amb ells. Creem els canals d'entrada i sortida amb les classes «DataInputStream» i «DataOutputStream» juntament amb els «Socket» i enviem un missatge de confirmació.

```
13     // Establim el port
14     int port = 2222;
15
16     // Creem el socket per a acceptar peticions
17     ServerSocket serverSocket = new ServerSocket(port);
18
19     // Creem el socket per a gestionar els missatges del client
20     Socket socketServei = serverSocket.accept();
21
22     // Creem els canals de Input i de Output
23     DataInputStream in = new DataInputStream(socketServei.getInputStream());
24     DataOutputStream out = new DataOutputStream(socketServei.getOutputStream());
25
26     // Enviam missatge confirmant la connexió
27     out.writeUTF("S'ha connectat correctament");
```



Per a calcular doncs, farem un bucle «do-while» del qual no es podrà sortir a no ser que el valor enviat sigui «=». Si el valor enviat és un nombre, el guardem en una variable apart, ja que si el que s'envia és un signe negatiu, caldrà recordar tal valor per poder operar.

```
29      //Rebem missatges i calculem
30      String missatge = "";
31      int nombre = 0;
32      int resultat = 0;
33      Boolean calcular = false;
34      do {
35          //Llegim el missatge
36          missatge = in.readUTF();
37          if (missatge.equals("=")) {
38              //Si és =, sortim
39              calcular = true;
40          } else if (missatge.equals("-")) {
41              //Si és -, restem l'últim nombre enviat
42              resultat = resultat - nombre;
43          } else {
44              //Si és un nombre, el sumem i guardem la variable per si es decideix restar
45              nombre = Integer.parseInt(missatge);
46              resultat = resultat + nombre;
47          }
48      }while (!calcular);
49
50      //Enviem el resultat final
51      out.writeUTF(Integer.toString(resultat));
```

Un cop calculat el resultat, l'enviem al client i tanquem els «sockets» juntament amb els canals d'entrada i sortida.

```
52
53      //Tanquem el socket
54      in.close();
55      out.close();
56      serverSocket.close();
57      socketServei.close();
58  }
59 }
```

## Client

El nom de la classe és **Ex1Client**.

```
1 package Madrenys_Helena_UF1_examen;
2
3 // @author: Helena Madrenys Planas
4
5 import java.io.DataInputStream;
6 import java.io.DataOutputStream;
7 import java.io.IOException;
8 import java.net.InetAddress;
9 import java.net.Socket;
10 import java.util.Scanner;
11
12 public class Ex1Client {
13     public static void main( String args[] ) throws IOException{
```

Cal obtenir la «IP» del servidor. Li podriem passar per teclat però com que estem treballant en local, podem simplement fer ús de la funció «InetAddress.getLocalHost()». A continuació creem el «Socket» amb aquesta «IP» i el port corresponent. Creem també els canals d'entrada i sortida.

```
14     // Establim el port i la IP del servidor
15     int port = 2222;
16     InetAddress serverHost = InetAddress.getLocalHost();
17
18     // Creem el socket per connectar-nos al servidor
19     Socket clientSocket = new Socket(serverHost, port);
20
21     // Creem els canals de Input i de Output
22     DataInputStream in = new DataInputStream(clientSocket.getInputStream());
23     DataOutputStream out = new DataOutputStream(clientSocket.getOutputStream());
24
25     // Comprovem la connexió
26     System.out.println(in.readUTF());
```

Ara fem un bucle «do-while» per anar enviant missatges mentre aquest no contingui el caràcter «=», aleshores es sortirà del bucle i es rebrà el resultat.

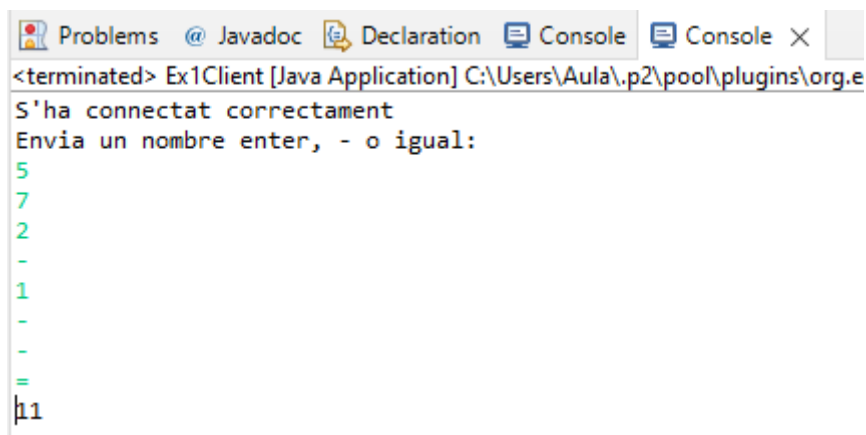


```
28 //Enviem operacions fins que el caràcter sigui =
29 System.out.println("Envia un nombre enter, - o igual:");
30 Scanner sc = new Scanner (System.in);
31 String enviar = "";
32 do {
33     enviar = sc.nextLine();
34     out.writeUTF(enviar);
35 }while (!enviar.equals("="));
36
37 //Rebem el resultat
38 System.out.println(in.readUTF());
39
```

Finalment tanquem el «Socket» i l' «Scanner» utilitzats juntament amb els canals d'entrada i sortida.

```
40 //Tanquem el socket
41 in.close();
42 out.close();
43 clientSocket.close();
44 sc.close();
45 }
46 }
```

## Output



```
Problems @ Javadoc Declaration Console Console X
<terminated> Ex1Client [Java Application] C:\Users\Aula\.p2\pool\plugins\org.e
S'ha connectat correctament
Envia un nombre enter, - o igual:
5
7
2
-
1
-
-
=
11
```



## Exercici 2

### Servidor

El nom de la classe és **Ex2Servidor**.

```
1 package Madrenys_Helena_UF1_examen;
2
3 //@author: Helena Madrenys Planas
4
5 import java.io.DataInputStream;
6 import java.io.DataOutputStream;
7 import java.io.IOException;
8 import java.net.ServerSocket;
9 import java.net.Socket;
10
11 public class Ex2Servidor {
12     public static void main( String args[] )throws IOException{
```

Per a crear la connexió fem el mateix que a Ex1Servidor.

```
13     //Establim el port
14     int port = 3333;
15
16     //Creem el socket per a acceptar peticions
17     ServerSocket serverSocket = new ServerSocket(port);
18
19     //Creem el socket per a gestionar els missatges del client
20     Socket socketServei = serverSocket.accept();
21
22     //Creem els canals de Input i de Output
23     DataInputStream in = new DataInputStream(socketServei.getInputStream());
24     DataOutputStream out = new DataOutputStream(socketServei.getOutputStream());
25
26     //Enviem missatge confirmant la connexió
27     out.writeUTF("S'ha connectat correctament");
```

Ara establim el mínim i el màxim per a poder fer ús de la funció «Math.random» i generar els dos nombres aleatòris. Seguidament els enviem al client.



```
29 //Diguem quin serà el valor màxim i quin serà el mínim
30 int min = 5;
31 int max = 20;
32
33 //Generem un número aleatori entre el min i el max
34 int random_int1 = (int)Math.floor(Math.random()*(max-min+1)+min);
35 int random_int2 = (int)Math.floor(Math.random()*(max-min+1)+min);
36
37 //Enviem els nombres al client
38 out.writeInt(random_int1);
39 out.writeInt(random_int2);
40 }
```

Rebem la resposta i comprovem si és correcte. Depenen de si ho és o no, enviem un missatge o un altre. Finalment tanquem els recursos.

```
41 //Rebem la resposta del client i comprovem
42 int resposta = in.readInt();
43 int resultat = random_int1 + random_int2;
44 if (resposta == resultat)
45 {
46     out.writeUTF("Resultat correcte");
47 } else
48 {
49     out.writeUTF("Resultat incorrecte, la resposta és " + Integer.toString(resultat));
50 }
51
52 //Tanquem el socket
53 in.close();
54 out.close();
55 serverSocket.close();
56 socketServei.close();
57 }
58 }
```

## Client

El nom de la classe és **Ex2Client**.

```
1 package Madrenys_Helena_UF1_examen;
2
3 //@author: Helena Madrenys Planas
4
5 import java.io.DataInputStream;
6 import java.io.DataOutputStream;
7 import java.io.IOException;
8 import java.net.InetAddress;
9 import java.net.Socket;
10 import java.util.Scanner;
11
12 public class Ex2Client {
13     public static void main( String args[] )throws IOException{
14         // ...
15     }
16 }
```

Creem la connexió com a Ex1Client.



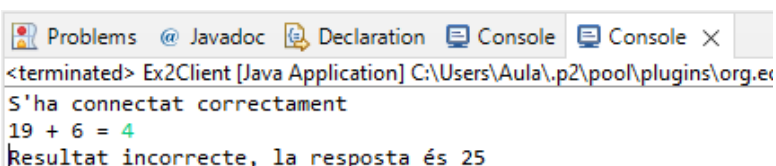
```
14 //Establim el port i la IP del servidor
15 int port = 3333;
16 InetAddress serverHost = InetAddress.getLocalHost();
17
18 //Creem el socket per connectar-nos al servidor
19 Socket clientSocket = new Socket(serverHost, port);
20
21 //Creem els canals de Input i de Output
22 DataInputStream in = new DataInputStream(clientSocket.getInputStream());
23 DataOutputStream out = new DataOutputStream(clientSocket.getOutputStream());
24
25 //Comprovem la connexió
26 System.out.println(in.readUTF());
```

Rebem els dos nombres i els mostrem per pantalla. A més, demanem la resposta del client i la enviem al servidor. Mostrem també la resposta que ens envia el servidor i finalment tanquem recursos.

```
28 //Rebem els dos nombres del servidor i enviem resposta
29 int num1 = in.readInt();
30 int num2 = in.readInt();
31 System.out.print(num1 + " + " + num2 + " = ");
32 Scanner sc = new Scanner (System.in);
33 int resposta = sc.nextInt();
34
35 //Enviem el resultat i rebem resposta del servidor
36 out.writeInt(resposta);
37 System.out.println(in.readUTF());
38
39 //Tanquem el socket
40 in.close();
41 out.close();
42 clientSocket.close();
43 sc.close();
44 }
45 }
```

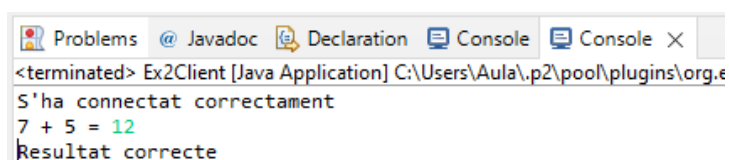
## Output

Resposta errònia:



```
<terminated> Ex2Client [Java Application] C:\Users\Aula\p2\pool\plugins\org.e
S'ha connectat correctament
19 + 6 = 4
Resultat incorrecte, la resposta és 25
```

Resposta correcte



```
<terminated> Ex2Client [Java Application] C:\Users\Aula\p2\pool\plugins\org.e
S'ha connectat correctament
7 + 5 = 12
Resultat correcte
```