Software Distribuït

SESSIÓ 3: Pràctica 1 - Client i Servidor

Execució del codi

Característiques definides per l'execució del codi:

servidor> java Server -h **(ha de mostrar un help)**
Us: java Server -p <port> [-i 1|2] **(ha de seguir aquest format en aquest ordre i detectar errors)**

client> java Client -h **(ha de mostrar un help)**
Us: java Client -s <maquina_servidora> -p <port> [-i 0|1] **(ha de seguir aquest format en aquest ordre i detectar errors)**

Llegir atentament les especificacions del projecte a realitzar penjades a la web de l'assignatura.

Proposta de control paràmetres d'entrada

```
HashMap<String,String> options = new HashMap();

for (int i=0; i<args.length; i=i+2)
{
      options.put(args[i],args[i+1]);
}

try{
      hostname = options.get("-s");
      port = Integer.parseInt(options.get("-p"));

      if (options.containsKey("-i")
      {
            //...//
      }

} catch{
      //...//
}
```

Estructuració del codi

Encapsulament i estructuració les funcions a realitzar:

- Ampliar ComUtils amb les funcions de baix nivell adequades al protocol (read_header, write_space, etc.).
- Funcions de més alt nivell lligades al protocol (tirar_daus, contar_punts, etc.)
- Classes i funcions lligades al funcionament del joc (màquina d'estats)
- Funcions de la lògica (get_winner, IA, etc.)

Intentar separar al màxim el protocol de la lògica del joc.

Disseny de la lògica (màquina d'estats)

C: STRT
S: CASH

C: BETT
S: LOOT
S: PLAY

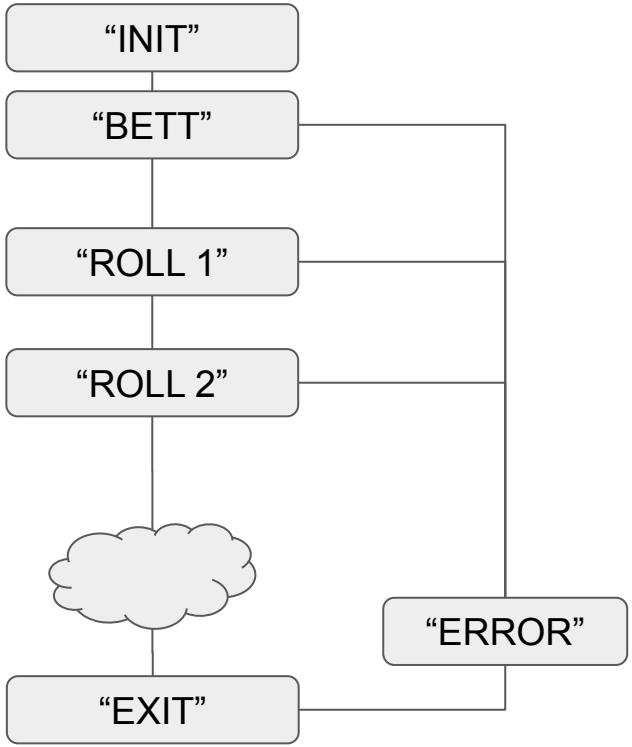
S: DICE
C: TAKE

S: DICE
C: TAKE

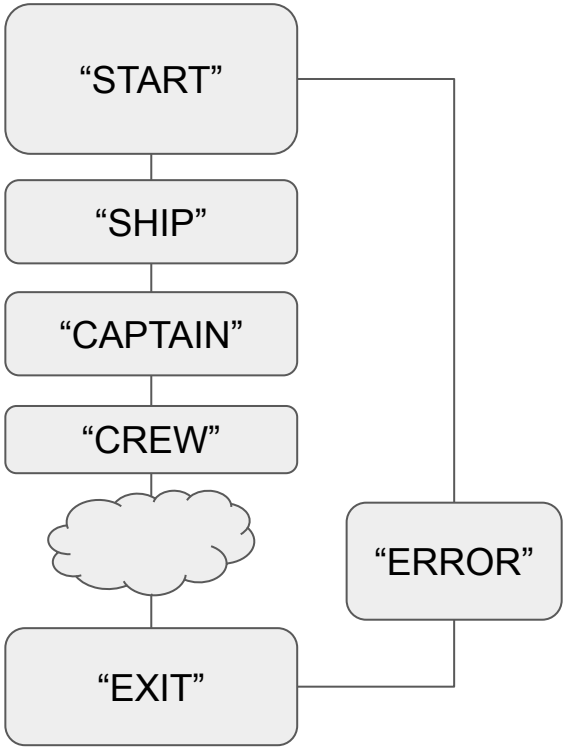
S: DICE
S: PNTS

...

EXEMPLE 1



EXEMPLE 2



Exemple Client

```
public class Client {

      public static void main(String[] args){

            //Tractament de paràmetres de consola

            try{
                  Socket socket = new Socket(nomMaquina, numPort);
                  socket.setSoTimeout(500*1000); //en ms.

            }  catch (IOException e) {
                  System.out.println("IOException: "+ e.getMessage());
            }
                        //...//
      }
}
```

Exemple Server

```
public class Server{

	public static void main(String[] args) throws IOException {
		//Tractament de paràmetres de consola

		try{
			ServerSocket serverSocket = new ServerSocket(numPort);
			Socket s = serverSocket.accept();
			s.setSoTimeout(500*1000);

			//...//

		}catch (IOException e) {
			System.out.println("IOException: "+ e.getMessage());
		}
	}
}
```

Exemple Server Multithread

```
public class ServerMT{
      public static void main(String[] args) throws IOException {
            //Tractament de paràmetres de consola
            try{
                  ServerSocket serverSocket = new ServerSocket(numPort);

                  while(true){
                        Socket s = serverSocket.accept();
                        s.setSoTimeout(500*1000); //en ms.
                        new Thread(new Game(s, var1, var2).start())
                  }

            }catch (IOException e) {
                  System.out.println("IOException: "+ e.getMessage());
            }
      }
}
```

Exemple Server Multithread de 2 jugadors

```
public class ServerMT2P{
      public static void main(String[] args) throws IOException {
            //Tractament de paràmetres de consola
            try{
                  ServerSocket serverSocket = new ServerSocket(numPort);

                  while(true){

                        /*Esperar a dos sockets disponibles*/
                              //...//

                        new Thread(new Game(s1,s2, var1, var2).start())
                  }

            }catch (IOException e) {
                  System.out.println("IOException: "+ e.getMessage());
            }
      }
}
```

ComUtils amb sockets

```
public ComUtils(InputStream inputStream, OutputStream outputStream) throws
IOException {

        dataInputStream = new DataInputStream(inputStream);
        dataOutputStream = new DataOutputStream(outputStream());

}



public ComUtils(Socket socket) throws IOException {

        /.../

        dataInputStream = new DataInputStream(socket.getInputStream());
        dataOutputStream = new DataOutputStream(socket.getOutputStream());

        /.../
}
```

Protocol

- EXERCICI: Definir les trames de error i les accions a dur a terme a través dels comentaris al repositoris corresponents de github.

- Possibles propostes de modificació o millora del protocol.

FI DEL CFR