IBERO
CIUDAD DE MÉXICO ®

TSU en Software

Práctica 9: Serialización y deserialización con Java y sockets

Desarrollo de Aplicaciones II

Profesor Omar Vázquez González

Carlos Araiza Dionicio

20-02-2021

Código

Código fuente:
https://github.com/iCharlieAraiza/DDAII/tree/main/Pr%C3%A1ctica%208%20y%209

Práctica 9A. Implementar lo siguiente:

Cliente:

- Solicitará al usuario los datos de una persona.
- Creará un objeto de clase "Persona" con los datos del usuario.
- Serializará el objeto usando XML (Manual)
- Enviará al Servidor el XML de la persona.

Servidor:

- Habilitará un socket de escucha.
- Recibirá el XML del cliente.
- Deserializará el XML enviado por el cliente (con una librería), y con ello instanciará un objeto "Persona".
- Guardará la información de la persona en una Base de Datos (MySQL).

Código del lado del servidor:

```
try{
    System.out.println("Esperando una conexión...");

    myClient = myServer.accept();
    System.out.println("Se ha aceptado la conexión.");

    InputStreamReader streamSocket = new
InputStreamReader(myClient.getInputStream());
    BufferedReader socketReader = new BufferedReader(streamSocket);
    PrintWriter socketWritter = new
PrintWriter(myClient.getOutputStream(),true);

    String message = socketReader.readLine();
    System.out.println("Mensaje recibido");
    System.out.println(message);

    Persona persona = new Persona();
    persona.fromXml(message);

    try{
        ConnectionDB db = new ConnectionDB();
        db.insertDatabase(persona);
    }catch (Exception e){
        System.out.println(e.getMessage());
    }

}catch (Exception e){
```

```
    System.out.println(e.getMessage());
}
```

Código del lado del cliente:

```
try{
   System.out.println("Conectando al servidor");
   myClient = new Socket(host, port);

   InputStreamReader streamSocket = new InputStreamReader(myClient.getInputStream());
   PrintWriter socketWriter =new PrintWriter( myClient.getOutputStream(), true);

   System.out.println(persona.toXML());

   socketWriter.println(persona.toXML());

}catch (Exception ex){
   System.out.println(ex);
}
```

Práctica 9B. Repetir lo anterior, utilizando serialización JSON.

- Cliente (Serialización Manual)
- Servidor (con librería)

Código del lado del servidor:

```
static private void practica9B(){
    try{
        System.out.println("Esperando una conexión...");

        myClient = myServer.accept();
        System.out.println("Se ha aceptado la conexión.");

        InputStreamReader streamSocket = new
InputStreamReader(myClient.getInputStream());
        BufferedReader socketReader = new BufferedReader(streamSocket);
        String message = socketReader.readLine();
        System.out.println("Mensaje recibido");
        System.out.println(message);
```

```
        Persona persona = new Persona();
        persona.fromJson(message);

        System.out.println( persona.toString() );

        try{
            ConnectionDB db = new ConnectionDB();
            db.insertDatabase(persona);
        }catch (Exception e){
            System.out.println(e.getMessage());
        }

    }catch (Exception e){
        System.out.println(e.getMessage());
    }
}
```

Código del lado del cliente:

```
try{
   System.out.println("Conectando al servidor");
   myClient = new Socket(host, port);


   InputStreamReader streamSocket = new InputStreamReader(myClient.getInputStream());
   PrintWriter socketWriter =new PrintWriter( myClient.getOutputStream(), true);


   socketWriter.println(persona.toJson());


}catch (Exception ex){
   System.out.println(ex);
}
```

Referencias

- Chat application using java sockets (with GUI). (2014, 17 diciembre). [Vídeo]. YouTube. https://www.youtube.com/watch?v=kqBmsLvWU14