



AE-1

Hilos y Sockets

SILVIAN VIRIGIL HOROIU,

MINERVA BARROSO MURILLO

ANTÓN MESÍAS CAMBA

SERGIO MARTÍNEZ PORTO

Requerimiento 1

```
import java.io.BufferedReader;

public class SocketServidor {

    public static final int PUERTO = 2020;

    public static void main(String[] args) {
        System.out.println("APLICACIÓN DE SERVIDOR");
        System.out.println("-----");

        List<Pelicula> peliculas = new ArrayList<>();

        peliculas.add(new Pelicula("A0215", "Encuentro en la tercera fase", "Steven Spielberg", 25));
        peliculas.add(new Pelicula("A025", "La lista de Schindler", "Steven Spielberg", 25));
        peliculas.add(new Pelicula("A752", "Los idiotas", "Lars Von Trier", 18));
        peliculas.add(new Pelicula("A962", "Cache", "Michael Haneke", 12));
        peliculas.add(new Pelicula("A0256", "Mulholland Drive", "David Lynch", 30));
    }
}
```

Menú que mostrará al cliente se añadirá en el SockerCliente

```
public static final int PUERTO = 2020;
public static final String IP_SERVER = "localhost";

public static void main(String[] args) {
    System.out.println("APLICACIÓN CLIENTE");
    System.out.println("MENÚ DE OPCIONES:");
    System.out.println("1. Consultar película por ID");
    System.out.println("2. Consultar película por título");
    System.out.println("3. Consultar películas por director");
    System.out.println("4. Salir de la aplicación");
    System.out.println("5. Añadir película");
    System.out.println("-----");

    try (Scanner userInput = new Scanner(System.in);
        Socket socketAlServidor = new Socket(IP_SERVER, PUERTO);
        PrintStream salida = new PrintStream(socketAlServidor.getOutputStream());
        InputStreamReader entrada = new InputStreamReader(socketAlServidor.getInputStream());
        BufferedReader bf = new BufferedReader(entrada)) {

        while (true) {
            System.out.print("Elija una opción: ");
            int option = userInput.nextInt();

            switch (option) {
                case 1:
                    // Consultar película por ID
                    break;
                case 2:
                    // Consultar película por título
                    System.out.print("Ingrese el título de la película: ");
                    String title = userInput.nextLine();
                    salida.println(option);
                    salida.println(title);
                    break;
            }
        }
    }
}
```

Parte del menú.

```
4  import java.io.InputStreamReader;
5  import java.io.PrintStream;
6  import java.net.Socket;
7  import java.util.Scanner;
8
9  public class SocketCliente {
10
11      public static final int PUERTO = 2020;
12      public static final String IP_SERVER = "localhost";
13
14  public static void main(String[] args) {
15      System.out.println("APLICACIÓN CLIENTE");
16      System.out.println("MENÚ DE OPCIONES:");
17      System.out.println("1. Consultar película por ID");
18      System.out.println("2. Consultar película por título");
19      System.out.println("3. Consultar películas por director");
20      System.out.println("4. Salir de la aplicación");
21      System.out.println("5. Añadir película");
22      System.out.println("-----");
23
24      try (Scanner userInput = new Scanner(System.in);
```

Markers Properties Servers Data Source Explorer Snippets Terminal Console ×

SocketServidor [Java Application] C:\Users\sergy\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.6.v20230121\jre\bin\java.exe

APLICACIÓN DE SERVIDOR

SERVIDOR: Esperando peticiones en el puerto 2020

SERVIDOR: Petición número 1 recibida

SERVIDOR: Me ha llegado del cliente: 2

SERVIDOR: Esperando peticiones en el puerto 2020

```

1 package cliente;
2 import java.io.BufferedReader;
3 import java.io.IOException;
4 import java.io.InputStreamReader;
5 import java.io.PrintStream;
6 import java.net.Socket;
7 import java.util.Scanner;
8
9 public class SocketCliente {
10
11     public static final int PUERTO = 2020;
12     public static final String IP_SERVER = "localhost";
13
14     public static void main(String[] args) {
15         System.out.println("APLICACIÓN CLIENTE");
16         System.out.println("MENÚ DE OPCIONES:");
17         System.out.println("1. Consultar película por ID");
18         System.out.println("2. Consultar película por título");
19         System.out.println("3. Consultar películas por director");
20         System.out.println("4. Salir de la aplicación");
21         System.out.println("5. Añadir película");
22         System.out.println("-----");
23
24         try (Scanner userInput = new Scanner(System.in);

```

Markers Properties Servers Data Source Explorer Snippets Terminal Console

SocketCliente [Java Application] C:\Users\sergy\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.6.v20230125-

APLICACIÓN CLIENTE
MENÚ DE OPCIONES:
1. Consultar película por ID
2. Consultar película por título
3. Consultar películas por director
4. Salir de la aplicación
5. Añadir película

Elija una opción: 2
Ingrese el título de la película: Elija una opción: |

Requerimiento 2

Consulta director. En este ejemplo le pasamos el nombre del director y nos pasa la película y el id.

```

24 películas.add(new Pelicula("A002", "Cache", "Michael Haneke", 12));
25 películas.add(new Pelicula("A0256", "Mulholland Drive", "David Lynch", 30));
26
27 try (ServerSocket serverSocket = new ServerSocket(PUERTO)) {
28     int peticion = 0;
29
30     while (true) {
31         System.out.println("SERVIDOR: Esperando peticiones en el puerto " + PUERTO);
32
33         try (Socket socketAlCliente = serverSocket.accept()) {
34             System.out.println("SERVIDOR: Petición número " + ++peticion + " recibida");
35
36             InputStreamReader entrada = new InputStreamReader(socketAlCliente.getInputStream());
37             BufferedReader bf = new BufferedReader(entrada);
38
39             String stringRecibido = bf.readLine();
40             System.out.println("SERVIDOR: Me ha llegado del cliente: " + stringRecibido);
41
42             if (stringRecibido.equals("1")) { // Search by ID
43                 String id = bf.readLine();
44                 Pelicula pelicula = buscarPeliculaPorID(id, películas);
45
46                 if (pelicula != null) {
47                     PrintStream salida = new PrintStream(socketAlCliente.getOutputStream());
48                     salida.println(pelicula.getId() + " " + pelicula.getTitulo() + " " + pelicula.getDirector() + " " + pelicula.getPrecio());
49                 }
50             }
51         }
52     }
53 }

```

Markers Properties Servers Data Source Explorer Snippets Terminal Console Console Console

SocketCliente [Java Application] C:\Users\sergy\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.6.v20230125-1136\jre\bin\javaw.exe (28 oct 2023 23:45:24) [pid: 18864]

1. Consultar película por ID
2. Consultar película por título
3. Consultar películas por director
4. Salir de la aplicación
5. Añadir película

Elija una opción: 3
Ingrese el nombre del director: David Lynch
ID: A0256 Título: Mulholland Drive Director: David Lynch Precio: 30
Elija una opción:

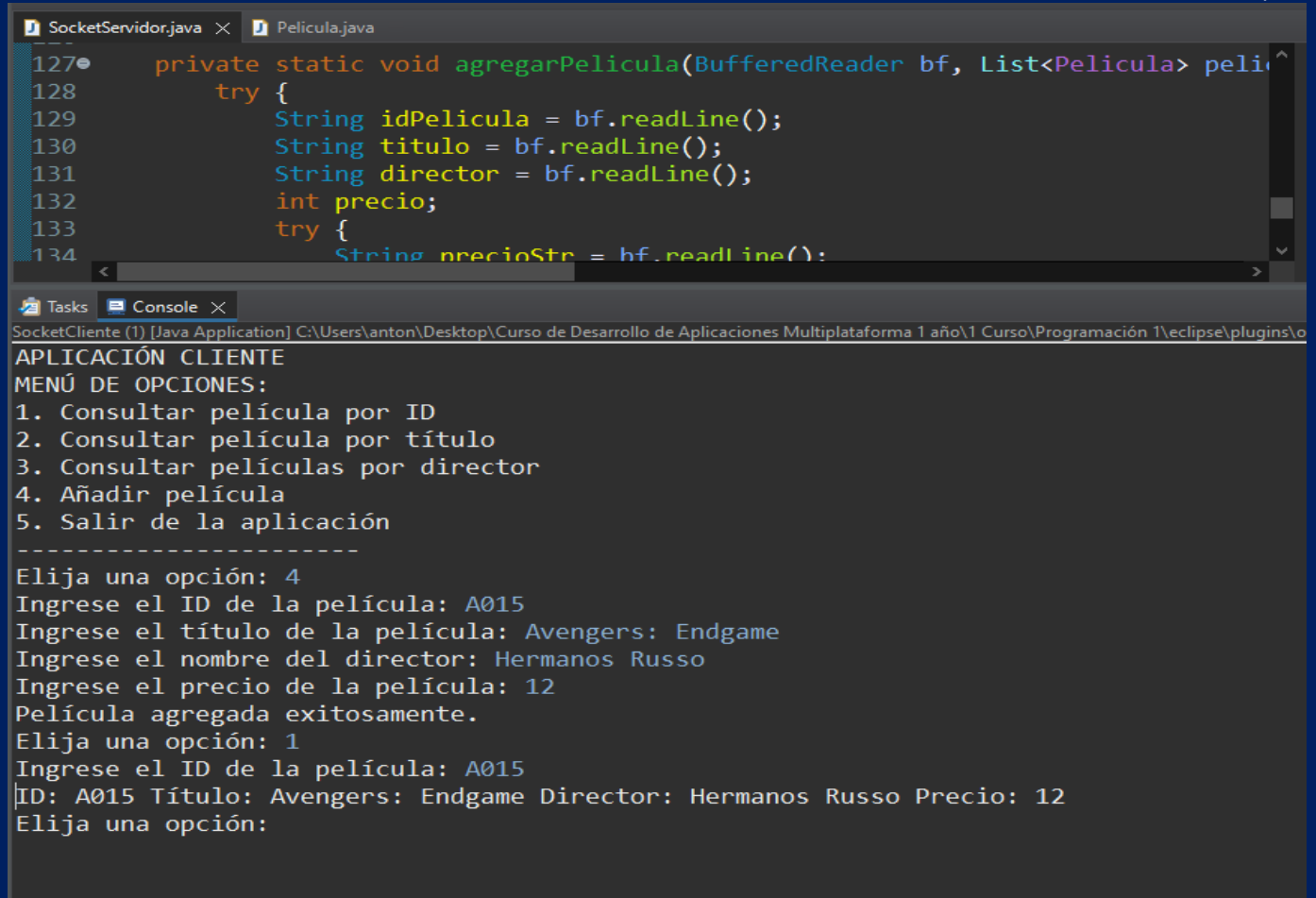
Requerimiento 3

Agregar película: en este método pedimos y recogemos los atributos de la clase Película y los añadimos a la lista de películas que tenemos.

```
126
127● private static void agregarPelícula(BufferedReader bf, List<Película> peliculas, Socket socketAlCliente) {
128     try {
129         String idPelícula = bf.readLine();
130         String titulo = bf.readLine();
131         String director = bf.readLine();
132         int precio;
133         try {
134             String precioStr = bf.readLine();
135             precio = Integer.parseInt(precioStr);
136         } catch (NumberFormatException e) {
137             System.err.println("SERVIDOR: Error al convertir precio a entero");
138             precio = 0; // Asigna un valor por defecto si el precio no es válido
139         }
140
141         synchronized (peliculas) {
142             peliculas.add(new Película(idPelícula, titulo, director, precio));
143         }
144
145         PrintStream salida = new PrintStream(socketAlCliente.getOutputStream());
146         salida.println("Película agregada exitosamente.");
147     } catch (IOException e) {
148         System.err.println("SERVIDOR: Error de entrada/salida al agregar película");
149         e.printStackTrace();
150     }
151 }
```

Aquí podemos comprobar el funcionamiento del método. Pedimos que se ingresen los datos por el teclado y, si los datos son correctos, devolvemos un mensaje de que la película se ha agregado correctamente. Si hubiese algún problema a la hora de agregar los datos, devolveríamos un mensaje de error.

Para comprobar que se ha añadido correctamente a nuestra Lista vamos a utilizar el método de buscar película por ID y el programa nos devuelve la película que hemos añadido anteriormente imprimiendo sus datos.



The screenshot shows the Eclipse IDE with two tabs: 'SocketServidor.java' and 'Pelicula.java'. The 'Pelicula.java' tab is active, displaying the following code:

```
127 private static void agregarPelicula(BufferedReader bf, List<Pelicula> peli
128     try {
129         String idPelicula = bf.readLine();
130         String titulo = bf.readLine();
131         String director = bf.readLine();
132         int precio;
133         try {
134             String precioStr = bf.readLine();
```

Below the code editor, the 'Console' tab is open, showing the output of a Java application. The output is as follows:

```
APLICACIÓN CLIENTE
MENÚ DE OPCIONES:
1. Consultar película por ID
2. Consultar película por título
3. Consultar películas por director
4. Añadir película
5. Salir de la aplicación
-----
Elija una opción: 4
Ingrese el ID de la película: A015
Ingrese el título de la película: Avengers: Endgame
Ingrese el nombre del director: Hermanos Russo
Ingrese el precio de la película: 12
Película agregada exitosamente.
Elija una opción: 1
Ingrese el ID de la película: A015
ID: A015 Título: Avengers: Endgame Director: Hermanos Russo Precio: 12
Elija una opción:
```

Metodología:

Para realizar este trabajo hemos decidido dividir el proyecto en partes para que así cada integrante del grupo aportara al proyecto y, utilizando el controlador de versiones Git, pudiéramos ir subiendo a nuestro repositorio de GitHub los avances que íbamos haciendo de la actividad.

Primero empezamos con la estructura y conexiones entre los sockets y por último fuimos implementando los métodos necesarios para el funcionamiento del proyecto.

Cabe destacar que en este proyecto hemos tenido alguna dificultad a la hora de que los sockets se comunicaran perfectamente entre sí, y también a la hora de implementar los métodos, sobre todo en el requerimiento 3, por lo tanto hemos tenido que utilizar foros y herramientas externas tales como:

- Stack Overflow
- La Web del Programador
- Chat GPT

GitHub:

<https://github.com/AntonSias/AE-1.-Hilos-y-Sockets>

