

## Introducción a la Programación de Sockets con UDP en Java

### ¿Qué son los Sockets?

Los sockets son una forma de establecer una comunicación entre procesos que se ejecutan en diferentes sistemas o en la misma máquina. Permiten que dos aplicaciones se comuniquen entre sí a través de una red, enviando y recibiendo datos.

### Protocolo UDP

UDP (User Datagram Protocol) es un protocolo de comunicación sin conexión, lo que significa que no se establece una conexión explícita antes de enviar datos. Es más ligero que TCP, pero no garantiza la entrega de los datos ni el orden en que se reciben.

### Programación de Sockets con UDP en Java

En Java, la programación de sockets con UDP es bastante sencilla. El paquete `java.net` proporciona las clases necesarias para trabajar con sockets.

### Creación de un Socket UDP

```
import java.net.*;

public class UDPServer {
    public static void main(String[] args) throws Exception {
        DatagramSocket serverSocket = new DatagramSocket(9876);

        byte[] receiveData = new byte[1024];

        while (true) {
            DatagramPacket receivePacket = new DatagramPacket(receiveData,
receiveData.length);
            serverSocket.receive(receivePacket);

            String mensaje = new String(receivePacket.getData());
            System.out.println("Mensaje recibido: " + mensaje);
        }
    }
}
```

### Envío de Datos mediante UDP

```
import java.net.*;

public class UDPClient {
    public static void main(String[] args) throws Exception {
        DatagramSocket clientSocket = new DatagramSocket();
        InetAddress IPAddress = InetAddress.getByName("localhost");
        byte[] sendData = "Hola desde el cliente".getBytes();
        DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, IPAddress,
9876);
        clientSocket.send(sendPacket);
        clientSocket.close();
    }
}
```

---

## Ejemplo Completo: Comunicación Cliente-Servidor

### Servidor UDP

```
import java.net.*;

public class UDPServer {
    public static void main(String[] args) throws Exception {
        DatagramSocket serverSocket = new DatagramSocket(9876);

        byte[] receiveData = new byte[1024];

        while (true) {
            DatagramPacket receivePacket = new DatagramPacket(receiveData,
receiveData.length);
            serverSocket.receive(receivePacket);

            String mensaje = new String(receivePacket.getData());
            System.out.println("Mensaje recibido: " + mensaje);
        }
    }
}
```

### Cliente UDP

```
import java.net.*;

public class UDPClient {
    public static void main(String[] args) throws Exception {
        DatagramSocket clientSocket = new DatagramSocket();

        InetAddress IPAddress = InetAddress.getByName("localhost");
        byte[] sendData = "Hola desde el cliente".getBytes();

        DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, IPAddress,
9876);
        clientSocket.send(sendPacket);

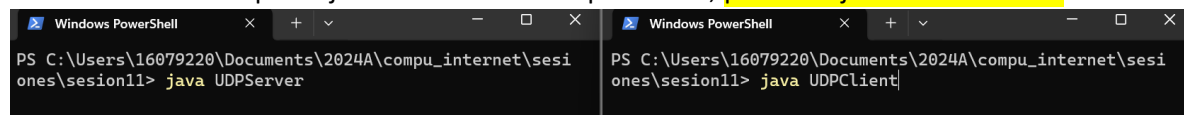
        clientSocket.close();
    }
}
```

**Guardar los archivos, compilar y ejecutar primero el servidor luego el cliente:**

Compilar

```
PS C:\Users\16079220\Documents\2024A\compu_internet\siones\sesion11> javac UDPServer.java
PS C:\Users\16079220\Documents\2024A\compu_internet\siones\sesion11> javac .\UDPClient.java
```

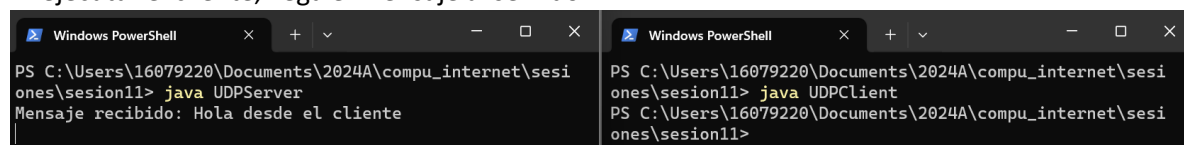
Usar dos terminales para ejecutar de forma independiente, **primero ejecutar el servidor.**



```
Windows PowerShell
PS C:\Users\16079220\Documents\2024A\compu_internet\siones\sesion11> java UDPServer

Windows PowerShell
PS C:\Users\16079220\Documents\2024A\compu_internet\siones\sesion11> java UDPClient
```

Al ejecutar el cliente, llega el mensaje al servidor:



```
Windows PowerShell
PS C:\Users\16079220\Documents\2024A\compu_internet\siones\sesion11> java UDPServer
Mensaje recibido: Hola desde el cliente

Windows PowerShell
PS C:\Users\16079220\Documents\2024A\compu_internet\siones\sesion11> java UDPClient
PS C:\Users\16079220\Documents\2024A\compu_internet\siones\sesion11>
```