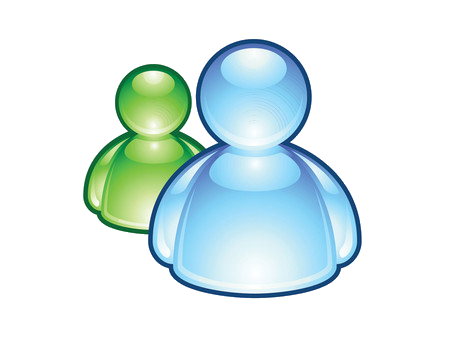


***Instituto Politecnico Nacional***

**Escuela Superior de Cómputo**

**Bocanegra Heziquio Yestlanezi   
  Practica 6  
  Sockets Servidores  
1. Chat 3D   
Fecha   
  2CM13  
  Programación Orientada a Objetos**



# Introducción

Los sockets son un mecanismo que nos permite establecer un enlace entre dos programas que se ejecutan independientes el uno del otro (generalmente un programa cliente y un programa servidor) Java por medio de la librería java.net nos provee dos clases: Socket para implementar la conexión desde el lado del cliente y ServerSocket que nos permitirá manipular la conexión desde el lado del servidor [1].

Antes de comenzar a ver código (que es lo que todos queremos) explicaré cómo será el funcionamiento de nuestra aplicación cliente servidor usando sockets en Java, cabe resaltar que tanto el cliente como el servidor no necesariamente deben estar implementados en Java, solo deben conocer sus direcciones IP y el puerto por el cual se comunicarán. Para nuestro ejemplo de sockets implementaremos ambos (cliente y servidor) usando Java y se comunicarán usando el puerto 1234 (es bueno elegir los puertos en el rango de 1024 hasta 65535) [1].

El servidor estará a la espera de una conexión, en cuanto el cliente inicie enviará un mensaje de petición al servidor, éste le responderá afirmativamente y una vez recibida la confirmación, el cliente enviará un par de mensajes y la conexión finalizará [1].

Como verás es un ejemplo simple, sin muchas complicaciones, así que manos a la obra, veamos el código [1]:

La clase Conexion simplemente nos brinda los datos que necesitados, mensajes de entrada, flujo de salida socket para el Cliente y socket para el servidor, estos últimos respectivamente inicializados desde el constructor. Las clases Cliente y Servidor que veremos en breve van a heredar de la clase Conexion para tener acceso a los atributos y a los sockets sin problemas [1].

## Clase servidor

La clase Servidor básicamente estará a la espera de que un cliente se conecte a él usando el socket en el puerto 1234, recibirá los mensajes, los mostrará y cerrará la conexión, es todo [1].

entonces al servidor que espera la conexión desde el cliente en el método accept(), una vez ésta sucede, envía un mensaje de confirmación con writeUTF("mensaje"), lee todos los mensajes enviados por el cliente con readLine() y cierra la conexión con el método close(). Veamos ahora la clase Cliente [1]

## Clase cliente

La clase Cliente, establecerá la conexión con el servidor usando un socket en localhost y el puerto 1234, una vez establece la conexión escribe dos mensajes en el servidor usando un ciclo for y cierra la conexión [1].

Vemos entonces que el cliente obtiene el flujo de salida de datos hacia el Servidor con el método getOutputStream y lo usa para enviarle un par de mensajes con el método writeUTF("mensaje..."), finalmente cierra la conexión con close() [1].

# DESARROLLO

# CONCLUSIÓN