

# Instituto Tecnológico de Morelia

Sistemas Operativos II

INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN Y  
COMUNICACIONES

## Programa Sockets Manual Técnico

PRESENTA(n):

**Castro Urieta Alondra 13121144 - nona.4londra@gmail.com**  
**Espinoza Sixtos Víctor Hugo 13121147 - bateriarosa@gmail.com**  
**Mogica Martínez Mariano 13121154 - marinhoted@gmail.com**  
**Pantoja Orozco Sabdi Abraham 13121157 - sabdi\_10@hotmail.com**

# ÍNDICE.

<b>INTRODUCCIÓN</b>	<b>2</b>
.....	
<b>OBJETIVO</b>	<b>3</b>
.....	
<b>OBJETIVOS ESPECÍFICOS</b>	<b>3</b>
.....	
<b>REQUISITOS DEL SISTEMA</b>	<b>3</b>
.....	
<b>HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO DEL PROGRAMA</b>	<b>4</b>
<b>Netbeans IDE</b>	<b>4</b>
.....	
<b>DIAGRAMA DE FLUJO DE LA EJECUCIÓN DEL PROGRAMA</b>	<b>5</b>
.....	
<b>DICCIONARIO DE CLASES</b>	<b>6</b>
.....	
<b>ELEMENTOS DE LA INTERFAZ GRÁFICA</b>	<b>9</b>
.....	
<b>GENERACIÓN DEL EJECUTABLE</b>	<b>10</b>
.....	

## INTRODUCCIÓN

El siguiente programa fue desarrollado para establecer un enlace en distintos equipos haciendo uso de Sockets. En este programa se establecerá una comunicación básica, en donde un equipo usará el módulo de servidor y el otro hará uso del módulo de cliente.

Este documento tiene como finalidad proporcionar al lector la lógica con la que ha sido desarrollado este programa, explicando los aspectos técnicos que hicieron posible la implementación del programa. Cabe mencionar que este documento no tiene como finalidad ser un curso de aprendizaje de los distintos usos de las diferentes herramientas que fueron utilizadas en el desarrollo del programa. Se recomienda consultar los manuales respectivos de cada una de las herramientas para un mayor detalle sobre ellos y su forma de operación.

## OBJETIVO

Mostrar los aspectos técnicos del programa desarrollado, así como ser una guía para el lector la cual le proveerá información necesaria para dar soporte y/o actualizar el programa.

## OBJETIVOS ESPECÍFICOS

- Mostrar el sistema.
- Definir los requerimientos de instalación y ejecución del programa.
- Relatar aspectos importantes a considerar en la instalación del programa.
- Mencionar aspectos del código (clases, funciones, variables, etc).
- Mostrar aspectos importantes del diseño del programa.
- Mostrar aspectos importantes durante la ejecución del programa así como la lógica del mismo.

## REQUISITOS DEL SISTEMA

Los siguientes requerimientos tanto en software como en hardware fueron de vital importancia para el desarrollo del programa y por ende serán necesarios en caso de una posterior actualización o modificación del código. Tomando como referencia que lo recomendado fue tanto el software como el hardware es lo que pide el IDE Netbeans.

ASPECTO	RECOMENDADO	MÍNIMO
SISTEMA OPERATIVO	Windows 10, Mac OS X 10.11.1	WINDOWS XP, Mac OS X 10.4.6
IDE	NETBEANS v8.1	NETBEANS 6.5.2
JAVA	v8 Update 91	Muy recomendable manejarla con lo recomendado.
PROCESADOR	Superior a 1.5 GHZ	
MEMORIA RAM	Superior a 2 GB.	

DISCO DURO	Superior a 1 GB
------------	-----------------

La conexión a internet no es necesaria salvo para la descarga del IDE.

## HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO DEL PROGRAMA

Para el desarrollo del programa de sockets se hizo uso de equipos que estuvieron dentro del rango de los requerimientos recomendados en el punto anterior, además que los equipos tuvieran Java instalado con el JDK en su versión 1.8.

Para la construcción del programa tanto de la parte de su funcionalidad así como la parte de su interfaz gráfica se hizo uso del entorno de desarrollo integrado (IDE) Netbeans en su versión 8.1 haciendo uso del lenguaje de programación Java.

### Netbeans IDE

Como ya fue mencionado en entorno de desarrollo utilizado fue Netbeans, el cual fue elegido por su gran facilidad de escribir, compilar, depurar y ejecutar programas, en nuestro caso utilizando el lenguaje de programación java. La descarga de este entorno se realizó en el sitio <https://netbeans.org/downloads/>, en donde actualmente se maneja la versión 8.1. Posterior a la descarga se realizó la instalación del entorno según lo indicado en el apartado del siguiente sitio <https://netbeans.org/community/releases/81/install.html>.

## DIAGRAMA DE FLUJO DE LA EJECUCIÓN DEL PROGRAMA.

El siguiente diagrama muestra la ejecución del programa, así como algunos estados del mismo.

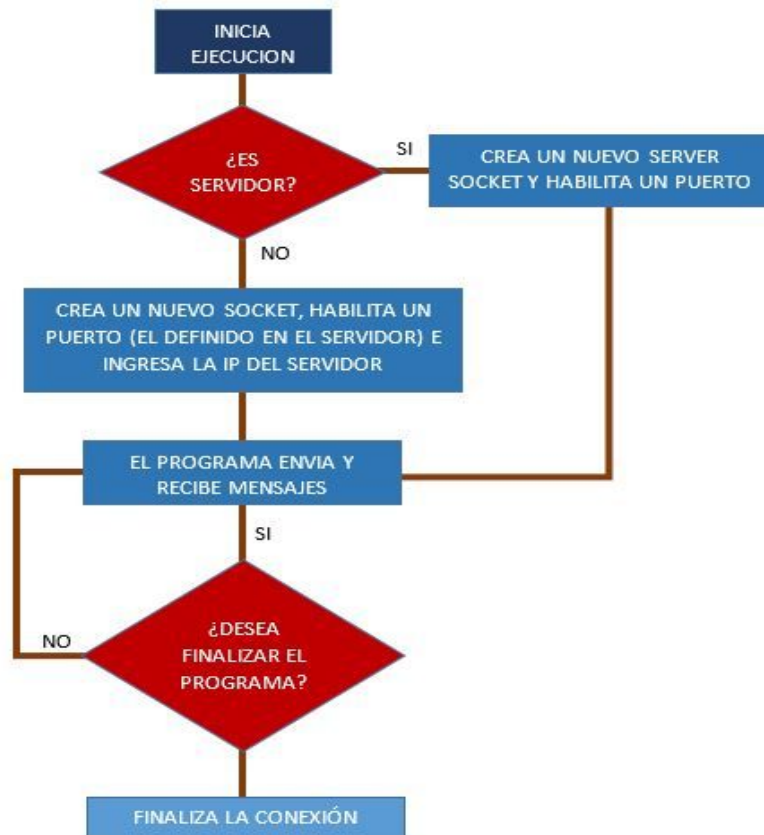


Figura 1.1 Diagrama de la ejecución del programa

La figura siguiente muestra la comunicación que se establece entre el cliente y el servidor.

## DICCIONARIO DE CLASES.

Las clases construidas y que usa el programa para su óptima ejecución son:

- **Conexion:** clase en la cual se realiza la conexión y la cual hace herencia de la clase thread, se inicializa el puerto y la ip en caso de que sea cliente y la necesite, posteriormente en su constructor crea un nuevo serverSocket y le indica el puerto de comunicación, en el caso de que sea cliente crea un nuevo socket en donde le indica el puerto y la ip del servidor.
- **Módulo:** Esta clase hereda de la clase padre Conexion, hace uso en el procedimiento run() y sendMsg() de los DataOutputStream (salida) y Bufferedader (entrada) de la clase conexión, con los cuales podremos enviar y recibir un mensaje, a su vez en el procedimiento run() se tiene un while que servirá para imprimir los mensajes que lleguen, el cual seguirá imprimiendo en caso de que el flujo de entrada sea diferente de null, también se hace uso de la clase calendar para colocar la hora correspondiente en la cual salio o llego el mensaje respectivamente, en esta misma clase se tiene el procedimiento closeConexion() que servirá para cerrar la conexión cuando el usuario lo desee.
- **interface:** clase que hereda de javax.swing.JFrame, en su constructor se deshabilitan los botones enviarBoton y closeS (ver sección de elementos de la interfaz gráfica), en el evento de su botón enviarBoton se hace uso del calendar para concatenar la hora del sistema con el mensaje que se enviará, el cual se obtiene del jTextField y este a su vez se limpia cuando se oprime dicho botón, en el evento del botón levantarServicio se crea un nuevo modulo y se inicia el servicio, a su vez se deshabilitan los elementos levantarServicio, opciones, levantarServicio, ipp y se habilitan enviarBoton, closeS, también se imprime en un JOptionPane un mensaje de que el servicio se inició correctamente. En el evento del botón closeS se tiene que se habilitan los elementos levantarServicio, opciones e ipp, mientras que se deshabilitan closeS y enviarBoton, tambien se imprime en un JOptionPane el mensaje de que el servicio se cerró correctamente. En el evento de opciones, se tiene que se mostrará el label jIP y el jTextField ipp en caso de que se seleccione "Cliente", caso contrario si se selecciona "Servidor", se ocultaran.

Las siguientes tablas se muestran detalle las clases, variables y procedimientos usados:

CLASE CONEXION		
Variables/Estructura	Tipo	Detalles
puerto	int	constante privada
host	String	constante privada
texto	String	variable protegida
ss	ServerSocket	variable protegida
cs	Socket	variable protegida
salida	DataOutputStream	variable protegida
entrada	BufferedReader	variable protegida
Procedimientos y Funciones		
Conexion		Constructor de la clase que recibe un String, el cual es comparado para ver si se crea la conexión en modo servidor, si es así se se crea el socket para el servidor con el parámetro puerto designado y se crea el socket para el cliente, si la conexión es en modo cliente se crea el socket para el cliente con los parámetros host y puerto.

CLASE MODULO		
Variables/Estructura	Tipo	Detalles
tiempo	String	
Procedimientos y Funciones		
Modulo		Constructor de la clase que recibe un String, el cual es enviado al constructor de la clase padre (Conexion en este caso) con



	la sentencia super.
run	Procedimiento para inicializar la conexión, ya sea en modo Servidor o en modo Cliente, comenzando el socket para el cliente (en modo servido), también se encarga de preparar el flujo de entrada y salida para enviar y recibir mensajes. Dentro de este procedimiento se tiene un while en el cual está a la espera de mensajes y los imprime al recibir alguno con la hora haciendo uso de la clase calendar de java.
sendMsg	Procedimiento que recibe un String el cual servirá para mandar un mensaje a través del flujo de salida.
closeConexion	Procedimiento que sirve para finalizar la conexión.

CLASE INTERFACE		
Variables/Estructura	Tipo	Detalles
tiempo	String	
cliente	Modulo	
Procedimientos y Funciones		
levantarServicioActionPerformed	Se crea un nuevo objeto modulo y se inicia el servicio, se bloquean los elementos de levantarServicio, opciones y ipp, mientras que se habilitan los elementos enviarBoton y closeS. Se crea un JOptionPane que manda un mensaje en pantalla sobre que el servicio se ha iniciado.	
enviarBotonActionPerformed	Se crea un objeto calendario, el cual servirá para imprimir la hora del sistema cuando se envíe un mensaje, se hace una llamada al procedimiento de sendMsg de la clase Modulo.	

closeSActionPerformed	Se cierra la conexión con el Módulo,, se habilitan los elementos levantarServicio, opciones y ipp, mientras que se deshabilitan elementos como closeS y enviarBoton, de igual manera se crea un JOptionPane donde se imprime que el servicio se cerró.
opcionesActionPerformed	Verifica si el elemento opciones está seleccionado como "Cliente" o como "Servidor", si esta como Servidor se ocultan los elementos ipp y jIP, caso contrario que sea cliente, los muestra.

## ELEMENTOS DE LA INTERFAZ GRÁFICA.

La clase interface a su vez es por la cual se puede tener la interacción entre el usuario y el programa, siendo dentro de la clase donde se tienen los siguientes componentes:

ID	NOMBRE DE VARIABLE	TIPO	DESCRIPCIÓN
1	opciones	JComboBox	Selecciona entre el servicio a seleccionar, si el programa actuará como cliente o como servidor.
2	levantarServicio	JButton	Sirve para iniciar el servicio (servidor o cliente), una vez oprimido inicia el envío y recepción de mensajes.
3	ipp	JTextField	Si en opciones se seleccionó cliente, es por donde el usuario ingresa la ip del servidor a conectarse.
4	jIP	JLabel	Etiqueta que sirve para mostrar un texto plano con la leyenda "Ingrese dirección ip:".
5	closeS	JButton	Para cerrar sesión con el cliente o servidor.
6	jArea	JTextArea	Área donde se mostrarán los mensajes enviados o recibidos por el equipo.
7	jTextField	JTextField	Para ingresar el mensaje a enviar.

8	enviarBoton	JButton	Botón para enviar lo que se escribió en el textField de al lado.
---	-------------	---------	--

De acuerdo con los ID de la tabla anterior los elementos de la interfaz gráfica se encuentran distribuidos de la siguiente manera:

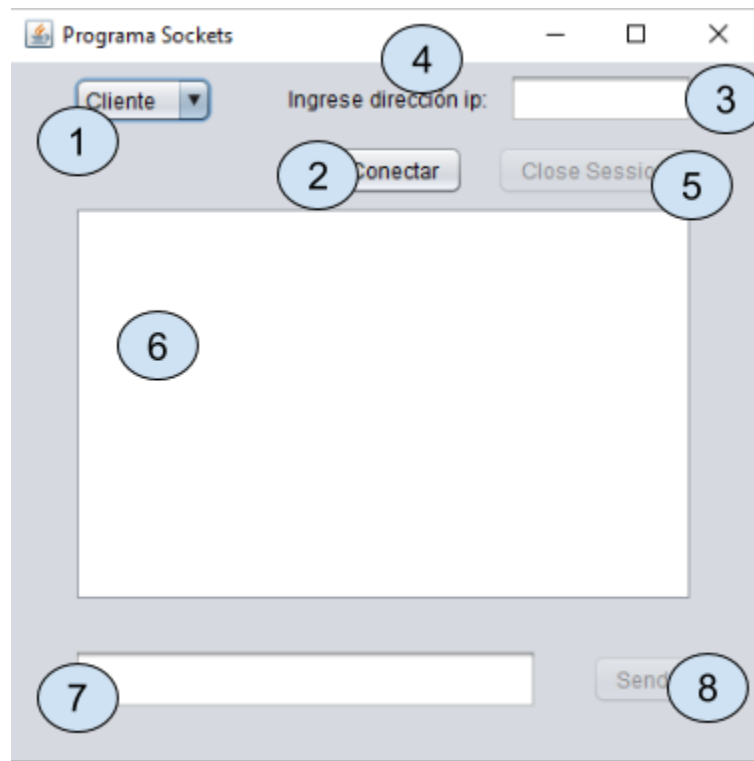
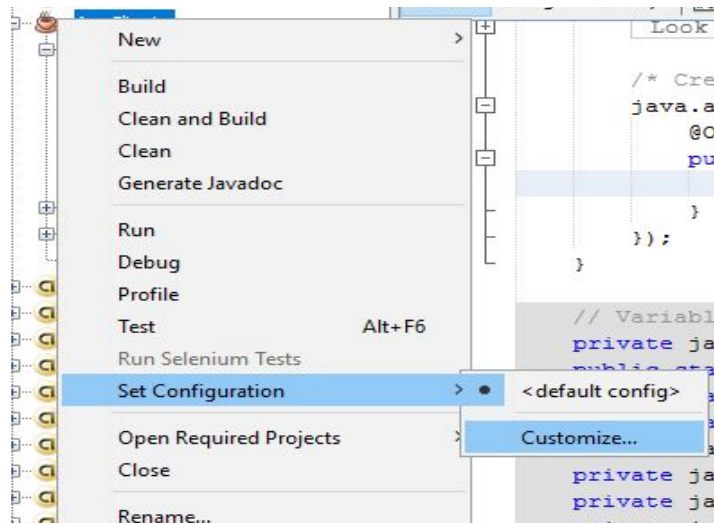


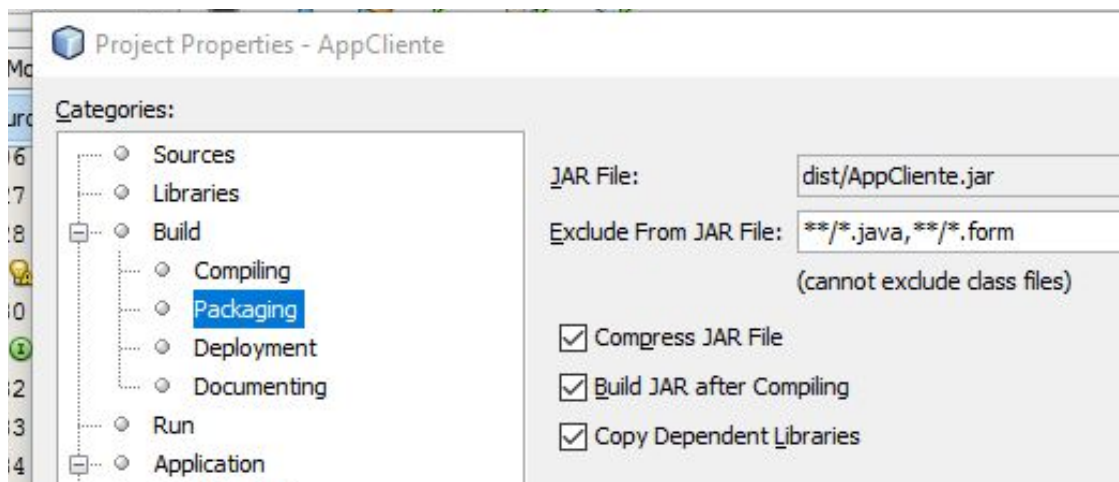
Figura 1.3 Interfaz gráfica

## GENERACIÓN DEL EJECUTABLE

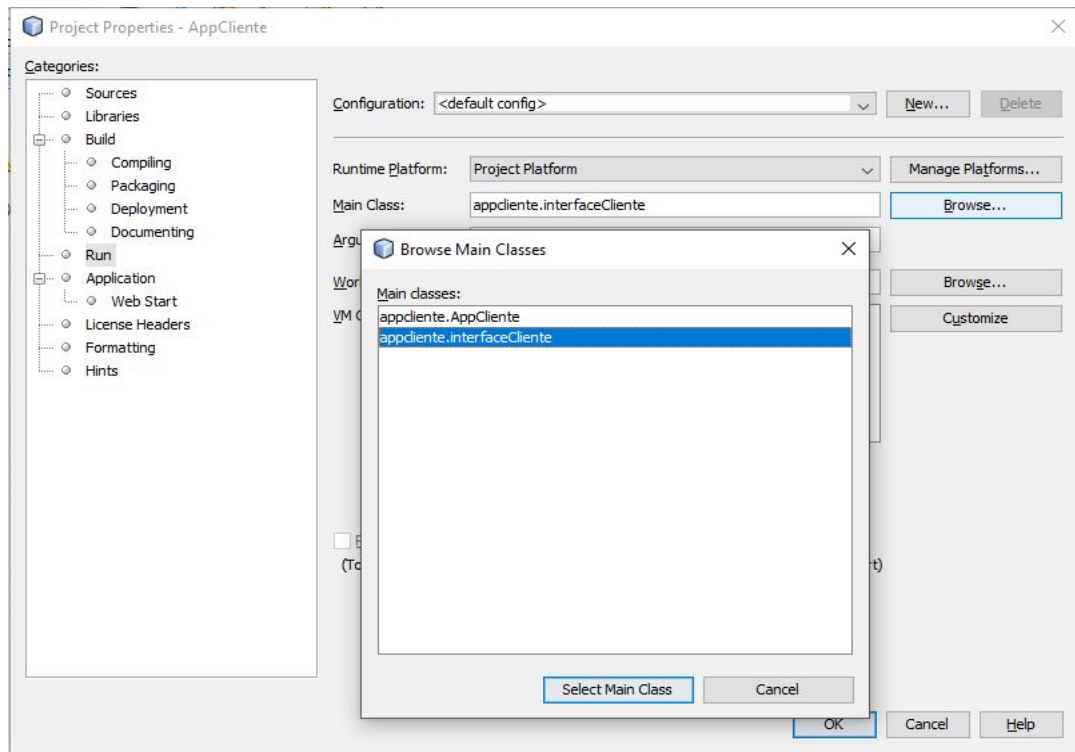
Para la generación del ejecutable en Netbeans solo se dió clic derecho en el proyecto creado, se seleccionó configuraciones (Set Configuration), después personalizar (customize).




Después de los pasos anteriores salió la ventana de propiedades del proyecto, en esta ventana nos fuimos a Packaging y seleccionamos la casilla Compress JAR File.



Dentro de la misma ventana nos fuimos a Run y seleccionamos como clase principal la ventana de la interfaz gráfica que es donde se comenzará la ejecución del programa.



Después de este paso se dio clic en ok en la ventana de propiedades del proyecto para guardar la configuración que se realizó. Como último paso se dió clic en la opción  (Clean and Build Project) para que se genera el archivo .jar.

Cabe mencionar que para ejecución del programa en cualquier equipo es necesario tener instalado java y ejecutar el archivo .jar.