**Descripción del equipo y de las herramientas de trabajo:**

## **Descripción del equipo de cómputo:**

Esta guía se desarrolló usando un computador de escritorio tipo portátil marca HP, con Windows 10 home, microprocesador Intel Core i5 de 1.8 GHZ 8a generación, 8 GB de memoria RAM unidad de almacenamiento de 250 GB de estado sólido, talla de 16.7 pulgadas, 2 gigas de memoria de vídeo tarjeta nvidia.



## **Descripción del software utilizado**

Para el desarrollo de esta guía fue necesario Descargar instalar y configurar adecuadamente las siguientes herramientas de software:

* Desarrollo de aplicaciones Java o JDK
  + Java SE Development Kit 8 - Downloads | Oracle Colombia
  + <https://www.oracle.com/co/java/technologies/javase/javase-jdk8-downloads.html>
  + INSTALACIÓN DEL JDK EN WINDOWS 10
  + <https://youtu.be/BoYRcKZbDb0>
* Entorno integrado de desarrollo en Netbeans 12.3
  + [I](https://youtu.be/oT1cUI984zU)NSTALACIÓN DE NETBEANS 12 WINDOWS
  + <https://youtu.be/oT1cUI984zU>
* Procesador de texto online Google Documents
* Code Block, complemento para Google Documents
* PlantUML Gizmo, complemento para Google Documents.

**Descripción de la aplicación tomada como ejemplo.**

Con el fin de simplificar el desarrollo de ejemplo y la explicación del mismo, he optado por realizar una simple aplicación tipo cliente servidor, la cual consta de dos sistemas:

Una aplicación que actúa como servicio y otra aplicación que actúa como cliente

Las características de la aplicación servidor Serán las siguientes:

* Inicia abriendo un puerto lógico de red específico estipulado por el usuario
* Acepta, atiende y procesa solicitudes de conexiones provenientes desde Una o varias instancias de aplicaciones clientes previamente desarrolladas para establecer comunicación con el servidor
* durante el proceso de conexión la aplicación servidor recibe 2 datos ( peso y altura) enviado por la aplicación cliente
* la aplicación servidor procesa estos datos y calcula el índice de masa corporal o IMC.
* Como respuesta el servidor envía el IMC y un mensaje a la aplicación cliente que realizó la solicitud.

Características de la aplicación cliente

* Al ejecutarla se despliega una interfaz de usuario con un formulario, el cual solicita al usuario ingresar la dirección IP y el número de puerto utilizados por la aplicación servidor
* El usuario un botón para conectarse con la aplicación servidor.
* como respuesta positiva de la conexión la aplicación despliega un formulario para ingresar los datos correspondientes al cálculo del IMC (peso y altura)
* el usuario pulsa un botón para enviar los datos al servidor el cual realiza el cálculo y devuelve el resultado con respuesta.
* La aplicación cliente muestra el resultado del cálculo enviado por el servidor.

A continuación se presenta boceto las interfaces de ambas aplicaciones:.

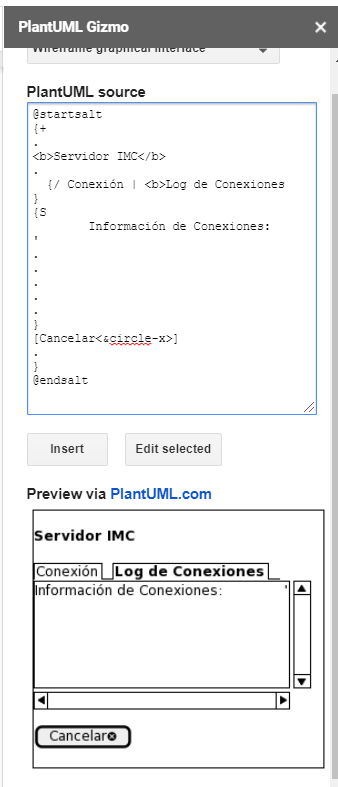
**GUÍA DE LA APLICACIÓN CLIENTE**

|  | @startsalt {+ . <b>Cliente IMC</b> .  {/ <b>Conexión | Cálculo IMC } .  {+  .  <I> IP Servidor: | " "  .  <I> Puerto Servidor: | " "  .  } . ..  {  [Cancelar<&circle-x>] |  [Conectar<&account-login>]   } } @endsalt |
| --- | --- |
|  | @startsalt {+ . <b>Servidor IMC .  {/ Conexion | <b>Calculo IMC }  == .  {  <I> PESO: | " " .  <I> ALTURA: | " " .   }  --  <B>Resultado: <I>0.0  ..  { . [Cancelar<&circle-x>] | [Cálculo<&account-login>]   } } @endsalt |

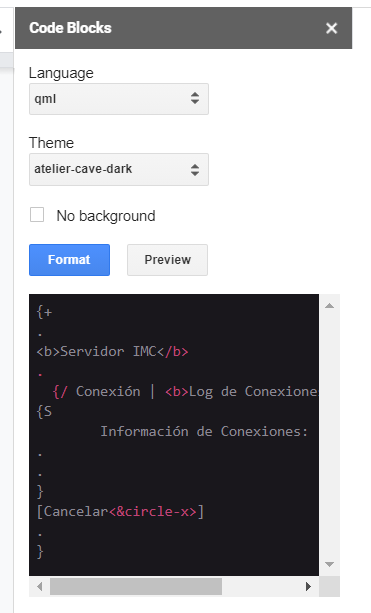
**GUÍA DE LA APLICACIÓN SERVIDOR**

|  | @startsalt {+ . Servidor IMC .  {/<b> Conexión | Log de Conexiones }  == . .  {  <B>Puerto de Servicio: | " "  .  }  --  .  <B>Estado: </b><I> Detenido  .  ..  {  .  [Cancelar<&circle-x>] |  [Iniciar <&account-login>]   } } @endsalt |
| --- | --- |
|  | @startsalt {+ . <b>Servidor IMC</b> .  {/ Conexión | <b>Log de Conexiones } {S  *Información de Conexiones: '*  . . . . . } [Cancelar<&circle-x>] . } @endsalt |

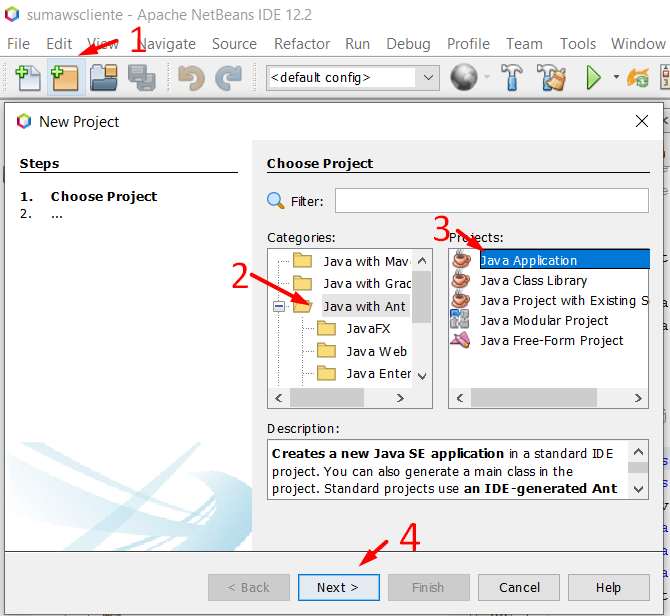
**Ejemplo el uso del complemento PlantUML GIZMO**



**Ejemplo del uso del complemento CODEBLOCKS**

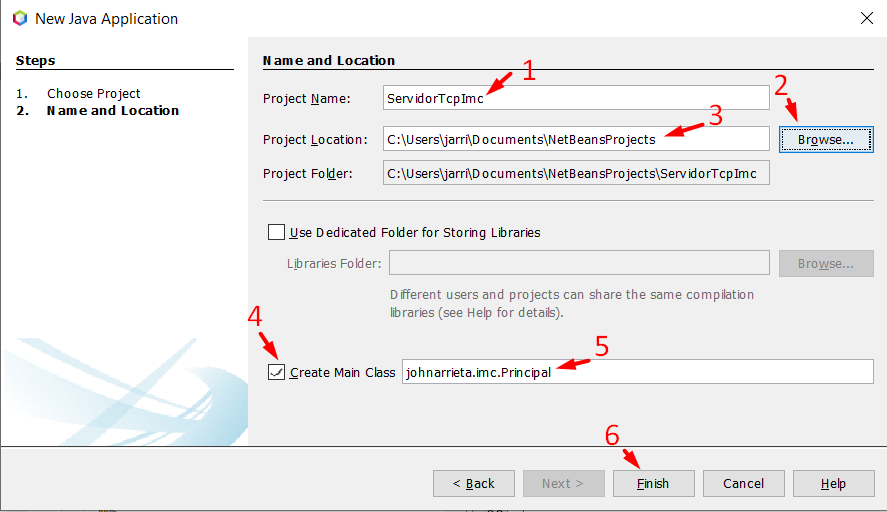


**CREAR EL PROYECTO EN NETBEANS**



Lo primero es crear el archivo:

1. Dar click en el botón nuevo proyecto.
2. Escoger la categoría java with ant.
3. y después elegir Java Application.
4. por último dar click en el botón next.



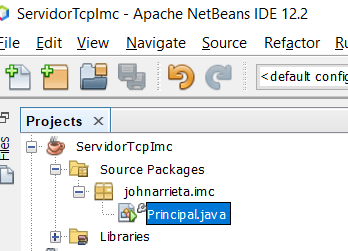
En esta ventana le ponemos nombre al proyecto esta es la parte del servidor:

1. En la casilla de project Names; colocamos el nombre del proyecto
2. y 3. damos click en el botón Browse y encontramos la carpeta donde queremos guardar el proyecto.

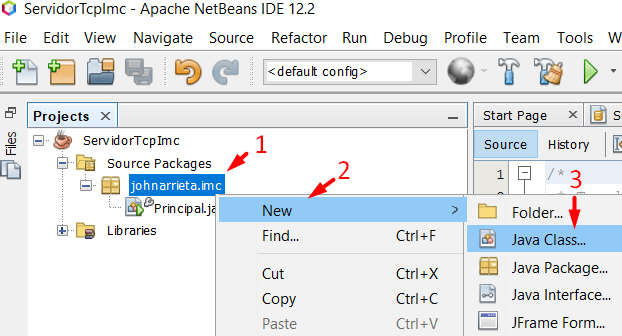
4. dar click en la casilla cuadrada y

5. al lado se coloca el nombre de la main class.

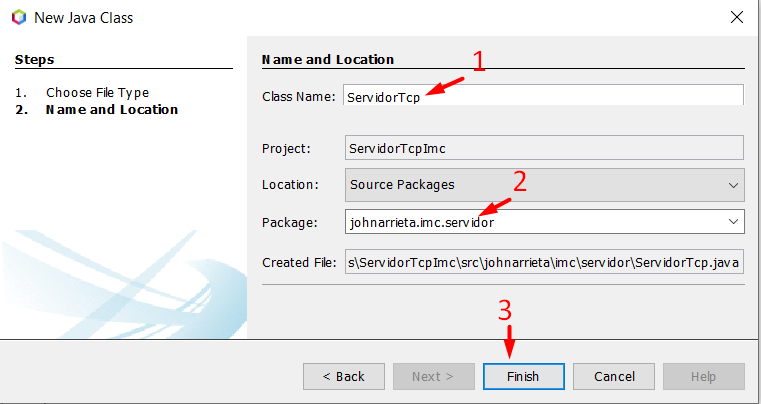
6. finalizamos todo dando click en el botón finish



verificamos si el paquete y la clase se creó correctamente

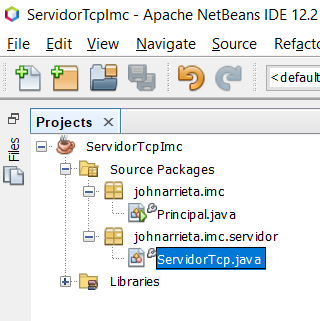


en el paquete recientemente creado le damos click derecho, saldrán diferente opciones de un menú y damos click en new y después click en java class

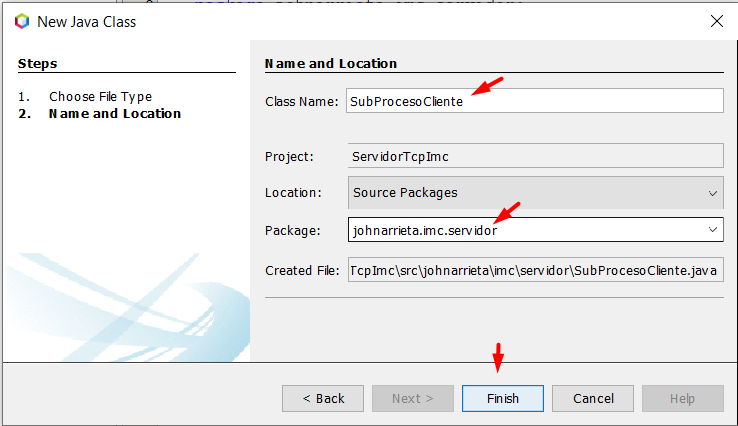


En esta ventana le ponemos nombre al proyecto esta es la parte del servidor:TCP

1. En la casilla de project Names; colocamos el nombre del proyecto servidor:TCP
2. En la casilla de package colocamos el nombre del paquete nuevo que se va crear.
3. finalizar todo dando click en el botón finish.



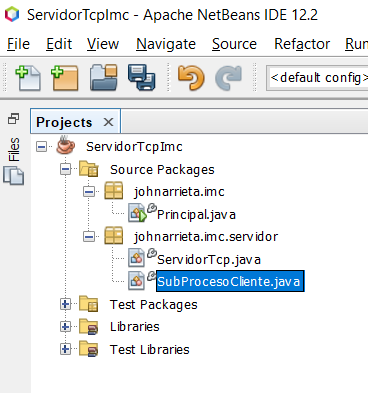
verificamos si el paquete y la clase se creó correctamente



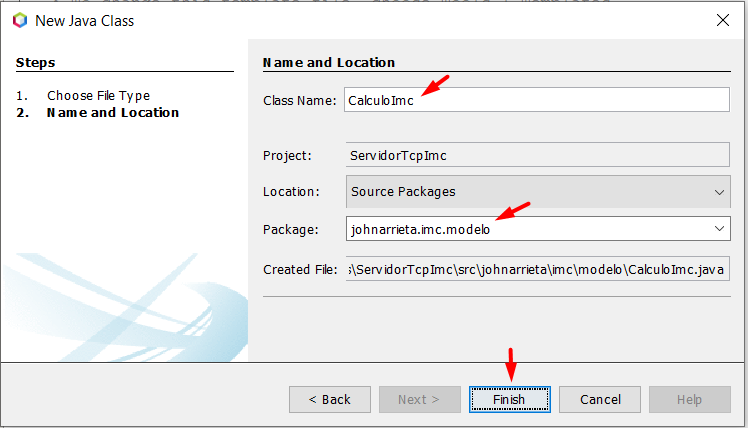
damos click derecho en un paquete, saldrá un menú con diferente opciones y damos click en new y después click en java class.

En esta ventana le ponemos nombre al proyecto esta es la parte del subProcesoCliente

1. En la casilla de project Names; colocamos el nombre del proyecto subProcesoCliente
2. En la casilla de package colocamos el nombre del paquete ya creado.
3. finalizar todo dando click en el botón finish.



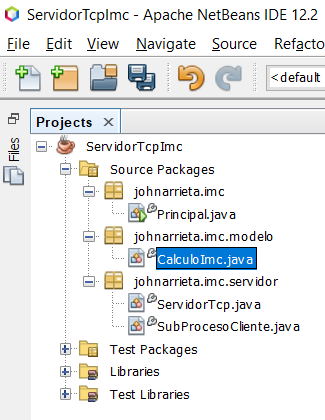
verificamos si el paquete y la clase se creó correctamente



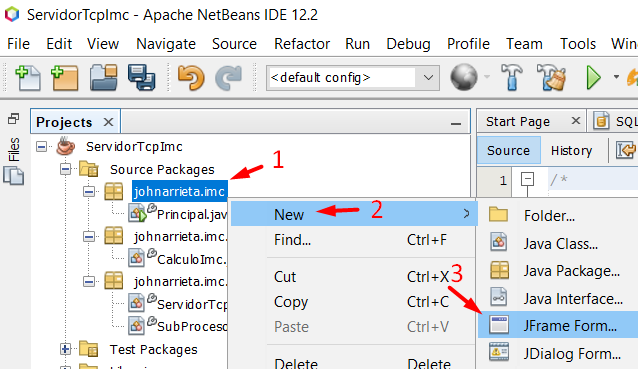
otra vez damos click derecho en un paquete, saldrá un menú con diferente opciones y damos click en new y después click en java class.

En esta ventana le ponemos nombre al proyecto esta es la parte del “CálculoImc”

1. En la casilla de project Names; colocamos el nombre del proyecto CálculoImc”
2. En la casilla de package colocamos el nombre del paquete nuevo que se va crear.
3. finalizar todo, dando click en el botón finish.

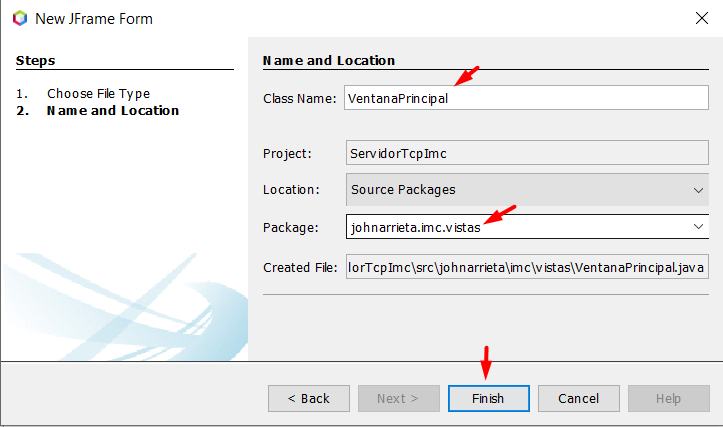


verificamos si el paquete y la clase se creó correctamente



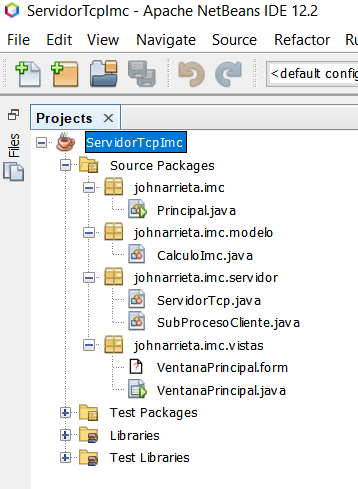
Damos click derecho en un paquete, saldrá un menú con diferente opciones y damos click en new y después click en jFrame Form.

.



En esta ventana le ponemos nombre al proyecto esta es la parte del “VentanaPrincipal”

1. En la casilla de project Names; colocamos el nombre del proyecto” VentanaPrincipal”
2. En la casilla de package colocamos el nombre del paquete nuevo que se va crear.
3. finalizar todo, dando click en el botón finish



Verificamos si el paquete y la clase se creó correctamente, confirmamos si creamos todos los archivos y paquetes necesario.

**ARCHIVO CALCULOIMC.JAVA**

| package johnarrieta.imc.modelo;  import java.io.Serializable;  */\*\*  \* @author JOHN CARLOS ARRIETA ARRIETA  \*/* public class CalculoImc implements Serializable {   private float peso;  private float altura;   static class Imc {   public float resultado;  public String mensaje;  }   private Imc imc;   public CalculoImc() { }   public CalculoImc(float peso, float altura) {  this.peso = peso;  this.altura = altura;  }   public Imc getImc() {  imc = new Imc();  if (peso <= 0 || altura <= 0) {  imc.mensaje = "ERROR: El peso y la altura deben ser mayores que 0";  return imc;  } else {  imc.resultado = peso / (altura \* altura);  if (imc.resultado < 18.5) {  imc.mensaje = "Debes consultar un Medico, tu peso es muy bajo";  } else if (imc.resultado >= 18.5 && imc.resultado <= 24.9) {  imc.mensaje = "Estas bien de peso";  } else if (imc.resultado > 24.9 && imc.resultado <= 29.9) {  imc.mensaje = "Debes bajar un poco de peso";  } else {  imc.mensaje = "Debes consultar un Medico, tu peso es muy alto";  }  return imc;  }   }   public float getPeso() {  return peso;  }   public void setPeso(float peso) {  this.peso = peso;  }   public float getAltura() {  return altura;  }   public void setAltura(float altura) {  this.altura = altura;  } } |
| --- |

Damos click en el archivo cálculoimc y creamos la clase, primero importamos Serializable y al lado del nombre de la clase ponemos “implements Serializable”.

A la clase de le agrega atributos como peso y altura, además se crea una clase llamada IMC que dentro de ella se tienen unos atributos llamados resultado y mensaje; después creamos un atributo u objeto con esa clase.

A la clase cálculoimc le añadimos 2 constructores; uno vacío y otro que acepte los atributos de peso y altura. además creamos un método llamado getImc el cual tiene la lógica para calcular el imc.

al final le creamos los setter y getter de los atributos de la clase.

**ARCHIVO SUBPROCESOCLIENTE.JAVA**

| package johnarrieta.imc.servidor;  import java.io.DataInputStream; import java.io.DataOutputStream; import java.io.IOException; import java.net.Socket; import java.text.SimpleDateFormat; import java.util.Date; import johnarrieta.imc.modelo.CalculoImc; import johnarrieta.imc.vistas.VentanaPrincipal;  */\*\*  \*  \* @author JOHN CARLOS ARRIETA ARRIETA  \*/* public class SubProcesoCliente extends Thread {   private Socket cliente;  private String ip;  private VentanaPrincipal ventana;     public SubProcesoCliente(Socket cliente, VentanaPrincipal v) {  this.cliente = cliente;  ip = cliente.getInetAddress().getHostAddress();  ventana = v;  }   @Override  public void run() {  try {  CalculoImc.Imc imc = calcularImc();  enviarRespuesta(imc);  } catch (Exception ex) {  System.out.println(log()+ex.getMessage());  ventana.getCajaLog().append(log()+ex.getMessage() + "\n");  try {  cliente.close();  } catch (IOException ex1) {  ServidorTcp.listaDeClientes.remove(ip);  } finally {  ServidorTcp.listaDeClientes.remove(ip);  }  }  }   public CalculoImc.Imc calcularImc() throws Exception {  DataInputStream input = null;  try {  input = new DataInputStream(cliente.getInputStream());  String msg = "Esperando el PESO: ";  System.out.println(log()+msg);  ventana.getCajaLog().append(log()+msg + "\n" + "\n");  float peso = input.readFloat();  msg = "PESO: " + peso;  System.out.println(log()+msg);  ventana.getCajaLog().append(log()+msg + "\n");  msg = "Esperando La Altura: ";  System.out.println(log()+msg);  ventana.getCajaLog().append(log()+msg + "\n");  float altura = input.readFloat();  msg = "ALTURA: " + altura;  System.out.println(log()+msg);  ventana.getCajaLog().append(log()+msg + "\n");  CalculoImc datosImc = new CalculoImc(peso, altura);  System.out.println(log()+"IMC: " + datosImc.getImc().resultado);  msg = "IMC: " + datosImc.getImc().resultado;  System.out.println(log()+msg);  ventana.getCajaLog().append(log()+msg + "\n");  System.out.println(log()+"MENSAJE: " + datosImc.getImc().mensaje);  msg = "MENSAJE: " + datosImc.getImc().mensaje;  System.out.println(log()+msg);  ventana.getCajaLog().append(log()+msg + "\n");  return datosImc.getImc();  } catch (IOException ex) {  String msg = "Error al caputurar datos del cliente " + ip;  System.out.println(log()+msg);  ventana.getCajaLog().append(log()+msg + "\n");  throw new Exception("Error al caputurar datos del cliente " + ip);   }   }   public void enviarRespuesta(CalculoImc.Imc imc) {   Thread hiloResponde = new Thread() {  @Override  public void run() {  DataOutputStream output = null;  try {  output = new DataOutputStream(cliente.getOutputStream());  output.writeFloat(imc.resultado);  output.writeUTF(imc.mensaje);  String msg = "IMC: " + imc.resultado;  System.out.println(log()+msg);  ventana.getCajaLog().append(log()+msg + "\n");  msg = "MENSAJE: " + imc.mensaje;  System.out.println(log()+msg);  ventana.getCajaLog().append(log()+msg + "\n");  output.flush();  enviarRespuesta(calcularImc());  } catch (IOException ex) {  String msg = "Error al enviar datos al cliente " + ip;  System.out.println(log()+msg);  ventana.getCajaLog().append(log()+msg + "\n");  ServidorTcp.listaDeClientes.remove(ip);  } catch (Exception ex) {  String msg = "Error al leer datos del cliente " + ip;  System.out.println(log()+msg);  ventana.getCajaLog().append(log()+msg + "\n");  try {  cliente.close();  } catch (IOException ex1) {  ServidorTcp.listaDeClientes.remove(ip);  } finally {  ServidorTcp.listaDeClientes.remove(ip);  }  }  }  };  hiloResponde.start();  }   public Socket getCliente() {  return cliente;  }   public void setCliente(Socket cliente) {  this.cliente = cliente;  }   public String log() {  SimpleDateFormat f = new SimpleDateFormat("dd-MM-yyyy hh:mm:ss a");  return ip + " -> " + f.format(new Date()) + " - ";  } } |
| --- |

Damos click en el archivo SubProcesoCliente y creamos la clase, primero importamos estas clases DataInputStream, DataOutputStream, IOException, Socket, SimpleDateFormat, Date, Cálculo imc, Ventana Principal. Se crea 3 atributos uno cliente, otro llamado ip y ventana, creamos un constructor que acepte solo la variable de tipo socket y ventana principal.

La clase se extiende de thread. Se agrega un override y debajo se crea un método llamado run el cual tienen un try que instancia un objeto de tipo cálculo imc.imc, ese objeto se mete dentro de un método, además tiene un catch que envía cualquier error ya sea por consola y también al log de conexiones. El catch también cierra la conexión con el cliente en específico.

Crear un método llamado calcularImc y al lado poner throws Exception, este método va a devolver un objeto de tipo Calcular.Imc, se crea un objeto de tipo DataInputStream, después hay un try catch dónde está la lógica del log del servidor donde aparecen todo lo que hace el cliente en la aplicación.

se crea otro método llamado enviarRespuesta se crea un objeto de tipo theader y un metodo run que tiene a su vez un try catch que envia el resultado imc al cliente.

**ARCHIVO SERVIDORTCP.JAVA**

| package johnarrieta.imc.servidor;  import java.awt.Color; import java.io.IOException; import java.net.ServerSocket; import java.net.Socket; import java.text.SimpleDateFormat; import java.util.Date; import java.util.HashMap; import java.util.Map; import java.util.function.Function; import johnarrieta.imc.vistas.VentanaPrincipal;  */\*\*  \* @author John Carlos Arrieta Arrieta  \*/* public class ServidorTcp extends Thread {   private Boolean estado;  public static Map<String, SubProcesoCliente> listaDeClientes;  private Integer puerto = 9007;  private ServerSocket servicio;  private VentanaPrincipal ventana;   public ServidorTcp(Integer puerto, VentanaPrincipal v) {  if (puerto != null || puerto != 0) {  this.puerto = puerto;  }  ventana = v;  listaDeClientes = new HashMap<>();  }   @Override  public void run() {  super.run(); *//To change body of generated methods, choose Tools | Templates.*  iniciarServicio();  }   public void iniciarServicio() {  try {  servicio = new ServerSocket(puerto);  estado = true;  ventana.getBtnIniciar().setText("DETENER");  ventana.getTxtEstado().setText("ONLINE");  ventana.getTxtEstado().setForeground(Color.green);  ventana.getBtnIniciar().setForeground(Color.RED);    String msg = log() + "Servidor disponible en el Puerto " + puerto;  System.out.println(msg);  ventana.getCajaLog().append(msg + "\n");  while (estado) {  Socket cliente = servicio.accept();  String ip = cliente.getInetAddress().getHostAddress();  msg =log()+"Cliente " + ip + " conectado";  System.out.println(msg + "\n");  ventana.getCajaLog().append(msg + "\n");  SubProcesoCliente atencion = new SubProcesoCliente(cliente, ventana);  ServidorTcp.listaDeClientes.put(ip, atencion);  atencion.start();  }  } catch (IOException ex) {  String msg = log()+"ERROR al abrir el puerto " + puerto;  System.out.println(msg);  ventana.getCajaLog().append(msg + "\n");  ventana.getBtnIniciar().setText("INICIAR");  ventana.getTxtEstado().setText("OFF LINE");  }  }   public void detenerServicio() {  if (estado) {  estado = false;  ventana.getBtnIniciar().setText("INICIAR");  ventana.getBtnIniciar().setForeground(Color.GREEN);  ventana.getTxtEstado().setText("OFF LINE");  ventana.getTxtEstado().setForeground(Color.RED);  ServidorTcp.listaDeClientes.entrySet().stream().map(new Function<Map.Entry<String, SubProcesoCliente>, String>() {  @Override  public String apply(Map.Entry<String, SubProcesoCliente> elemento) {  String ip = elemento.getKey();  SubProcesoCliente cliente = elemento.getValue();  String msg = log()+"Desconectando cliente " + ip;  System.out.println(msg);  ventana.getCajaLog().append(msg + "\n");  try {  cliente.getCliente().close();  cliente = null;  ServidorTcp.listaDeClientes.remove(elemento);  msg = log()+"Cliente desconectado" + ip;  System.out.println(msg);  ventana.getCajaLog().append(msg + "\n");  } catch (IOException ex) {  cliente = null;  ServidorTcp.listaDeClientes.remove(elemento);  msg = log()+"Cliente desconectado" + ip;  System.out.println(msg);  ventana.getCajaLog().append(msg + "\n");  }  return ip;  }  }).forEachOrdered(ip -> {  System.out.println("cliente " + ip + " Desconectado");  });  try {  servicio.close();  } catch (IOException ex) {  System.out.println("ERRRO no se puede cerrar el Puerto " + puerto);  String msg = log()+"ERRRO no se puede cerrar el Puerto " + puerto;  System.out.println(msg);  ventana.getCajaLog().append(msg + "\n");  }  }  }   public String log() {  SimpleDateFormat f = new SimpleDateFormat("dd-MM-yyyy hh:mm:ss a");  return f.format(new Date()) + " - ";  } } |
| --- |

Damos click en el archivo servidortcp, se importa Color; IOException; ServerSocket; Socket; SimpleDateFormat; Date; HashMap; Map; Function; VentanaPrincipal. se extiende se la clase thread, los atributos de la clase son de tipo boolean llamado estado, un Map de tipo String, SubProcesoCliente con el nombre listaDeClientes, Integer con el nombre puerto , private ServerSocket con el nombre servicio; Ventana Principal con el nombre ventana.

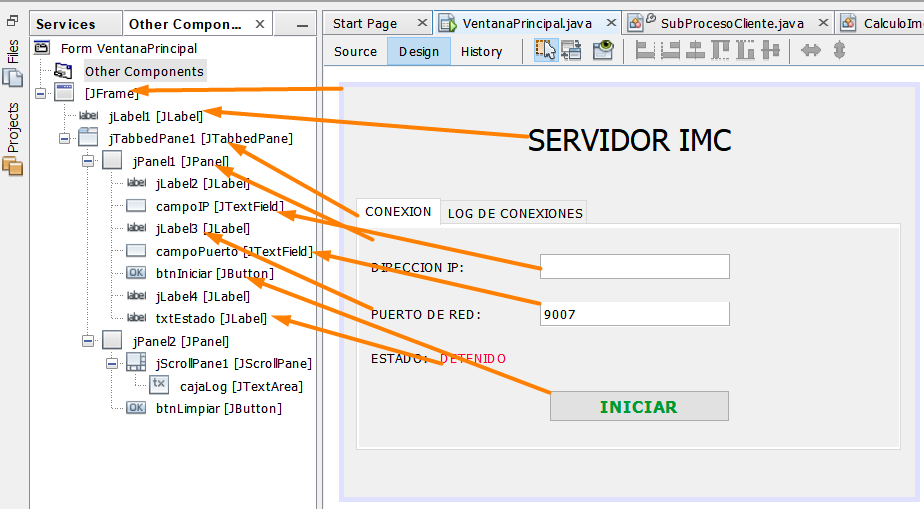
un constructor que acepta solo la ventana principal y el puerto.

se crea un método run y adentro se llama un método de la clase padre con el super que también se llama run además se llama al método iniciar servicio.

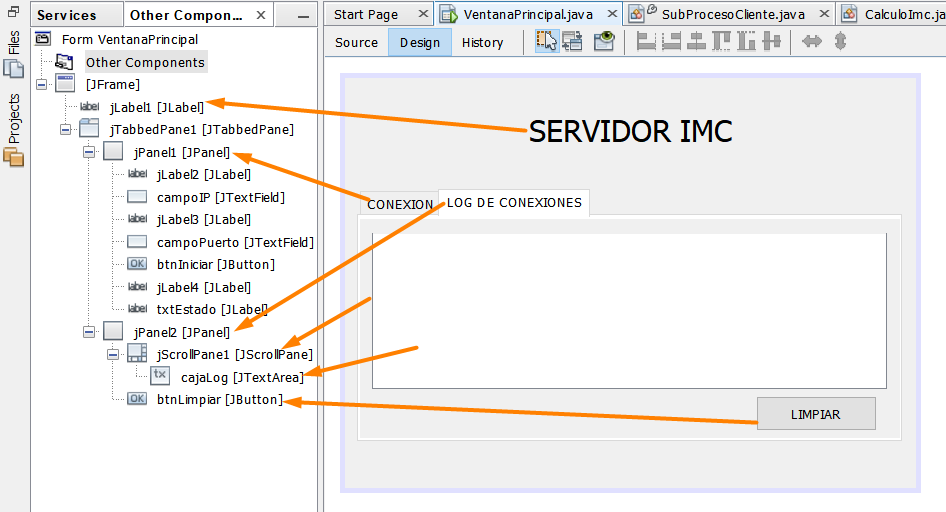
se crea un método iniciar servicio que inicia el servicio y conecta el servicio con el puerto, además de de conectar a los clientes y avisar que s e conectó el cliente.

hay otro método que se llama detener Servicio que detiene toda la coneccion con el cliente y el puerto.

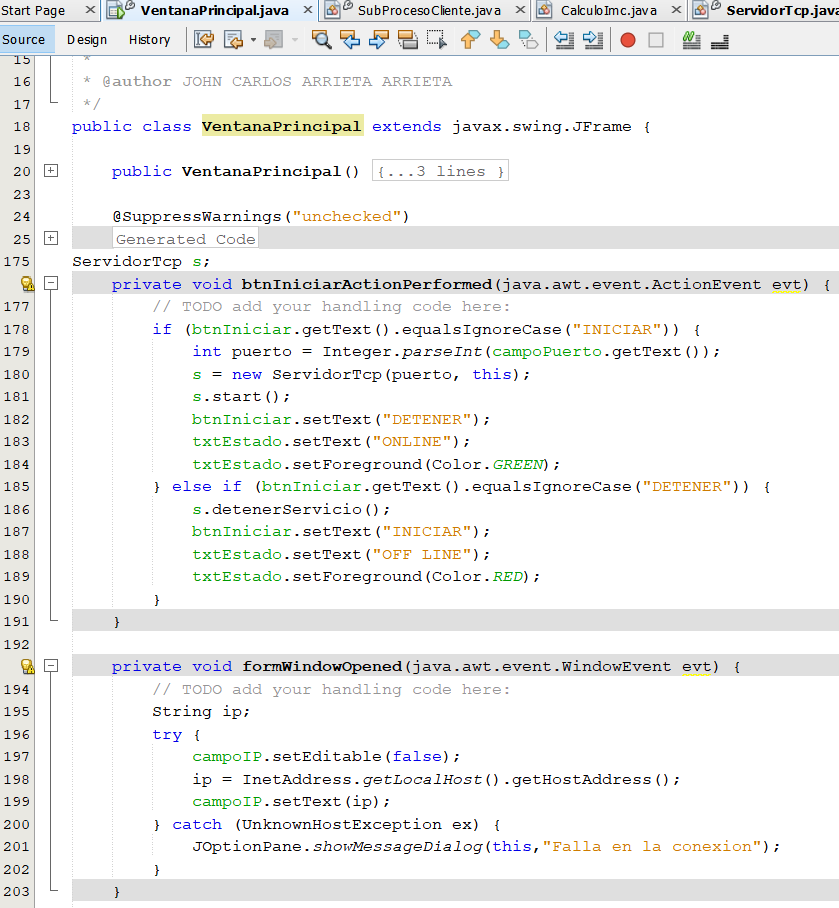
**ARCHIVO VENTANAPRINCIPAL.JAVA**



En el archivo ventana principal del servidor, creamos la ventana de la parte del servidor con un jtable creamos el nombre “servidor IMC”, debajo ponemos un tabbedpane; dentro de él ponemos 2 panel, en el primer panel llamado conexión ponemos dos textfield; uno para dirección ip y otro para el puerto (cada textfield tiene su respectivo jlabel) además dos jlabel; uno para el estado y otro para la respuesta del estado, al final con un botón con el nombre Iniciar.

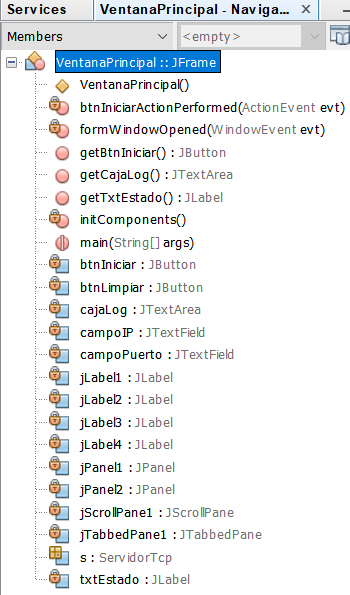


en el otro panel lo llamamos log de conexión y adentro se pone un scrollpane, abajo un botón con el nombre limpiar.



Dar click en el archivo ventana principal, afuera del método crea una variable de tipo servidor tcp en la en el método del botón iniciar se pone un if el cual se activa se el botón iniciar tiene el texto iniciar, crea una variable puerto que obtiene el puerto de la casilla puerto, crea un objeto de tipo servidor tcp y le encía el puerto y asi mismo, se llama también el método start del servidor tcp, cambia los textos del botón y txtestado(también cambia su color).

En el método de abajo el form, se crea una variable ip, y le pone la dirección ip del host address. y lo muestra en la casilla ip.



se rectifica que todo este creado como la imagen.

| package johnarrieta.imc.vistas;  import java.awt.Color; import java.net.InetAddress; import java.net.UnknownHostException; import java.util.logging.Level; import java.util.logging.Logger; import javax.swing.JButton; import javax.swing.JLabel; import javax.swing.JOptionPane; import javax.swing.JTextArea; import johnarrieta.imc.servidor.ServidorTcp;  */\*\*  \*  \* @author JOHN CARLOS ARRIETA ARRIETA  \*/* public class VentanaPrincipal extends javax.swing.JFrame {   public VentanaPrincipal() {  initComponents();  }   @SuppressWarnings("unchecked")  *// <editor-fold defaultstate="collapsed" desc="Generated Code">*   private void initComponents() {   jLabel1 = new javax.swing.JLabel();  jTabbedPane1 = new javax.swing.JTabbedPane();  jPanel1 = new javax.swing.JPanel();  jLabel2 = new javax.swing.JLabel();  campoIP = new javax.swing.JTextField();  jLabel3 = new javax.swing.JLabel();  campoPuerto = new javax.swing.JTextField();  btnIniciar = new javax.swing.JButton();  jLabel4 = new javax.swing.JLabel();  txtEstado = new javax.swing.JLabel();  jPanel2 = new javax.swing.JPanel();  jScrollPane1 = new javax.swing.JScrollPane();  cajaLog = new javax.swing.JTextArea();  btnLimpiar = new javax.swing.JButton();   setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);  addWindowListener(new java.awt.event.WindowAdapter() {  public void windowOpened(java.awt.event.WindowEvent evt) {  formWindowOpened(evt);  }  });   jLabel1.setFont(new java.awt.Font("Tahoma", 0, 24)); *// NOI18N*  jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);  jLabel1.setText("SERVIDOR IMC");   jLabel2.setText("DIRECCION IP: ");   jLabel3.setText("PUERTO DE RED:");   campoPuerto.setText("9007");   btnIniciar.setFont(new java.awt.Font("Tahoma", 1, 14)); *// NOI18N*  btnIniciar.setForeground(new java.awt.Color(0, 153, 51));  btnIniciar.setText("INICIAR");  btnIniciar.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  btnIniciarActionPerformed(evt);  }  });   jLabel4.setText("ESTADO: ");   txtEstado.setForeground(new java.awt.Color(255, 0, 51));  txtEstado.setText("DETENIDO");   javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);  jPanel1.setLayout(jPanel1Layout);  jPanel1Layout.setHorizontalGroup(  jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(jPanel1Layout.createSequentialGroup()  .addContainerGap()  .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)  .addGroup(jPanel1Layout.createSequentialGroup()  .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED\_SIZE, 131, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  .addComponent(campoIP, javax.swing.GroupLayout.PREFERRED\_SIZE, 152, javax.swing.GroupLayout.PREFERRED\_SIZE))  .addGroup(jPanel1Layout.createSequentialGroup()  .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED\_SIZE, 131, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  .addComponent(campoPuerto))  .addGroup(jPanel1Layout.createSequentialGroup()  .addComponent(jLabel4)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  .addComponent(txtEstado, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))  .addComponent(btnIniciar, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.PREFERRED\_SIZE, 145, javax.swing.GroupLayout.PREFERRED\_SIZE))  .addContainerGap(135, Short.MAX\_VALUE))  );  jPanel1Layout.setVerticalGroup(  jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(jPanel1Layout.createSequentialGroup()  .addGap(21, 21, 21)  .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  .addComponent(jLabel2)  .addComponent(campoIP, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))  .addGap(18, 18, 18)  .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  .addComponent(jLabel3)  .addComponent(campoPuerto, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))  .addGap(18, 18, 18)  .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  .addComponent(jLabel4)  .addComponent(txtEstado))  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 19, Short.MAX\_VALUE)  .addComponent(btnIniciar)  .addGap(20, 20, 20))  );   jTabbedPane1.addTab("CONEXION", jPanel1);   cajaLog.setEditable(false);  cajaLog.setColumns(20);  cajaLog.setRows(5);  jScrollPane1.setViewportView(cajaLog);   btnLimpiar.setText("LIMPIAR");   javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);  jPanel2.setLayout(jPanel2Layout);  jPanel2Layout.setHorizontalGroup(  jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel2Layout.createSequentialGroup()  .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)  .addGroup(jPanel2Layout.createSequentialGroup()  .addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)  .addComponent(btnLimpiar, javax.swing.GroupLayout.PREFERRED\_SIZE, 97, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(8, 8, 8))  .addGroup(jPanel2Layout.createSequentialGroup()  .addContainerGap()  .addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT\_SIZE, 412, Short.MAX\_VALUE)))  .addContainerGap())  );  jPanel2Layout.setVerticalGroup(  jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(jPanel2Layout.createSequentialGroup()  .addContainerGap()  .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 125, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  .addComponent(btnLimpiar, javax.swing.GroupLayout.DEFAULT\_SIZE, 28, Short.MAX\_VALUE)  .addGap(5, 5, 5))  );   jTabbedPane1.addTab("LOG DE CONEXIONES", jPanel2);   javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());  getContentPane().setLayout(layout);  layout.setHorizontalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(layout.createSequentialGroup()  .addContainerGap()  .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 437, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jTabbedPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))  .addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))  );  layout.setVerticalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(layout.createSequentialGroup()  .addGap(27, 27, 27)  .addComponent(jLabel1)  .addGap(32, 32, 32)  .addComponent(jTabbedPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 203, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addContainerGap(37, Short.MAX\_VALUE))  );   pack();  }*// </editor-fold>*  ServidorTcp s;  private void btnIniciarActionPerformed(java.awt.event.ActionEvent evt) {   *// TODO add your handling code here:*  if (btnIniciar.getText().equalsIgnoreCase("INICIAR")) {  int puerto = Integer.parseInt(campoPuerto.getText());  s = new ServidorTcp(puerto, this);  s.start();  btnIniciar.setText("DETENER");  txtEstado.setText("ONLINE");  txtEstado.setForeground(Color.GREEN);  } else if (btnIniciar.getText().equalsIgnoreCase("DETENER")) {  s.detenerServicio();  btnIniciar.setText("INICIAR");  txtEstado.setText("OFF LINE");  txtEstado.setForeground(Color.RED);  }  }    private void formWindowOpened(java.awt.event.WindowEvent evt) {   *// TODO add your handling code here:*  String ip;  try {  campoIP.setEditable(false);  ip = InetAddress.getLocalHost().getHostAddress();  campoIP.setText(ip);  } catch (UnknownHostException ex) {  JOptionPane.showMessageDialog(this,"Falla en la conexion");  }  }    public JLabel getTxtEstado() {  return txtEstado;  }   public JTextArea getCajaLog() {  return cajaLog;  }   public JButton getBtnIniciar() {  return btnIniciar;  }   */\*\*  \* @param args the command line arguments  \*/*  public static void main(String args[]) {  */\* Set the Nimbus look and feel \*/*  *//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">*  */\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.  \* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html   \*/*  try {  for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {  if ("Nimbus".equals(info.getName())) {  javax.swing.UIManager.setLookAndFeel(info.getClassName());  break;  }  }  } catch (ClassNotFoundException ex) {  java.util.logging.Logger.getLogger(VentanaPrincipal.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (InstantiationException ex) {  java.util.logging.Logger.getLogger(VentanaPrincipal.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (IllegalAccessException ex) {  java.util.logging.Logger.getLogger(VentanaPrincipal.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (javax.swing.UnsupportedLookAndFeelException ex) {  java.util.logging.Logger.getLogger(VentanaPrincipal.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  }  *//</editor-fold>*   */\* Create and display the form \*/*  java.awt.EventQueue.invokeLater(new Runnable() {  public void run() {  new VentanaPrincipal().setVisible(true);  }  });  }   *// Variables declaration - do not modify*   private javax.swing.JButton btnIniciar;  private javax.swing.JButton btnLimpiar;  private javax.swing.JTextArea cajaLog;  private javax.swing.JTextField campoIP;  private javax.swing.JTextField campoPuerto;  private javax.swing.JLabel jLabel1;  private javax.swing.JLabel jLabel2;  private javax.swing.JLabel jLabel3;  private javax.swing.JLabel jLabel4;  private javax.swing.JPanel jPanel1;  private javax.swing.JPanel jPanel2;  private javax.swing.JScrollPane jScrollPane1;  private javax.swing.JTabbedPane jTabbedPane1;  private javax.swing.JLabel txtEstado;  *// End of variables declaration*  } |
| --- |

se importan estas clases Color; InetAddress;.UnknownHostException; Level; Logger; JButton;JLabel;JOptionPane;JTextArea; ServidorTcp; Dar click en el archivo ventana principal, afuera del método crea una variable de tipo servidor tcp en la en el método del botón iniciar se pone un if el cual se activa se el botón iniciar tiene el texto iniciar, crea una variable puerto que obtiene el puerto de la casilla puerto, crea un objeto de tipo servidor tcp y le encía el puerto y asi mismo, se llama también el método start del servidor tcp, cambia los textos del botón y txtestado(también cambia su color).

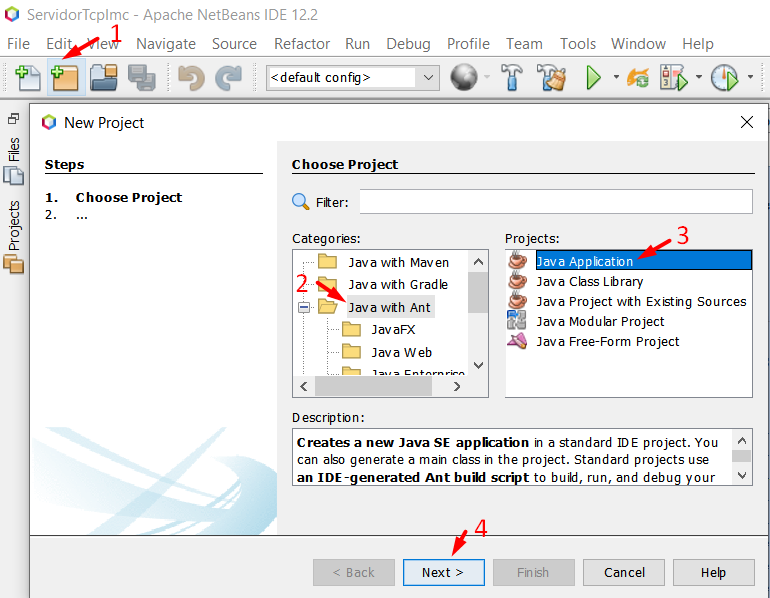
En el método de abajo el form, se crea una variable ip, y le pone la dirección ip del host address. y lo muestra en la casilla ip.

ARCHIVO PRINCIPAL.JAVA

| package johnarrieta.imc;  import johnarrieta.imc.vistas.VentanaPrincipal;  */\*\*  \* @author JOHN CARLOS ARRIETA ARRIETA  \*/* public class Principal {   */\*\*  \* @param args the command line arguments  \*/*  public static void main(String[] args) {  VentanaPrincipal v = new VentanaPrincipal();  v.setLocationRelativeTo(null);  v.setVisible(true);  } } |
| --- |

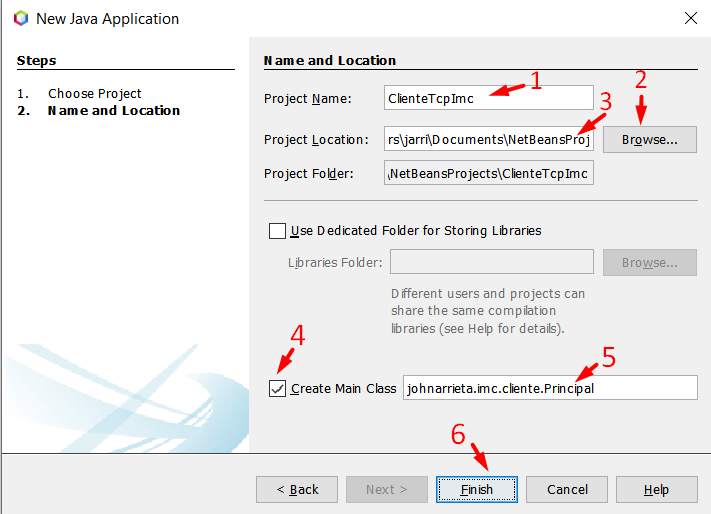
Se crea en la main class la instancia de la ventana principal, y se muestra.

**LA APLIACION CLIENTE**



Lo primero es crear el nuevo archivo para el cliente :

1. Dar click en el botón nuevo proyecto.
2. Escoger la categoría java with ant.
3. y después elegir Java Application.
4. por último dar click en el botón next.



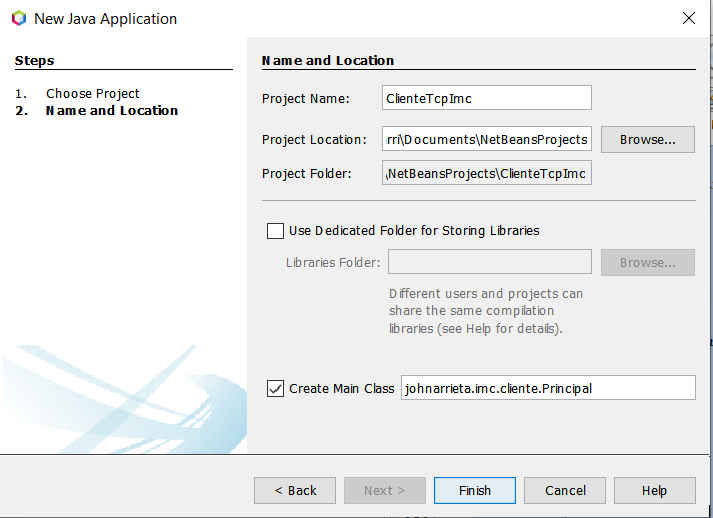
En esta ventana le ponemos nombre al proyecto esta es la parte del cliente:

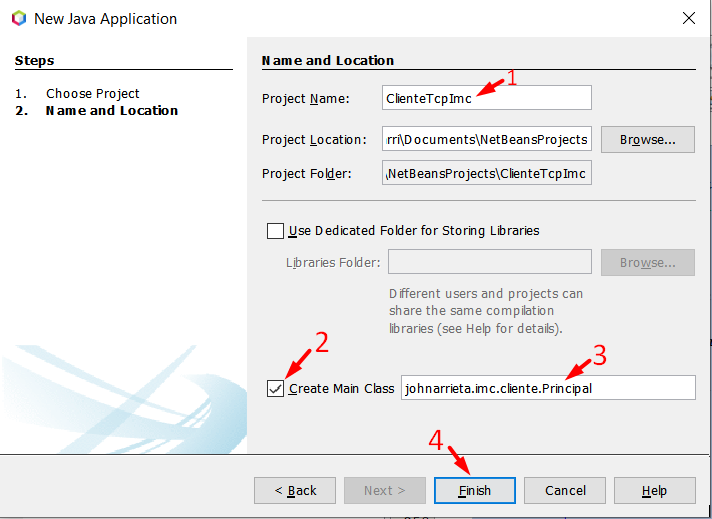
1. En la casilla de project Names; colocamos el nombre del proyecto
2. y 3. damos click en el botón Browse y encontramos la carpeta donde queremos guardar el proyecto.

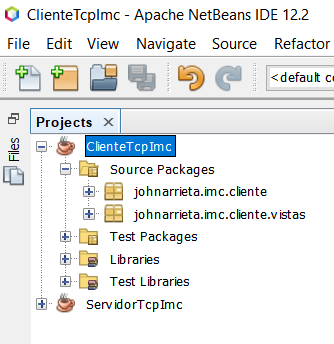
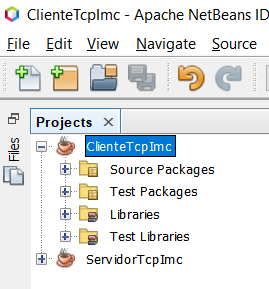
4. dar click en la casilla cuadrada y

5. al lado se coloca el nombre de la main class cliente.

6. finalizamos todo dando click en el botón finish







En esta ventana le ponemos nombre al proyecto esta es la parte del cliente:

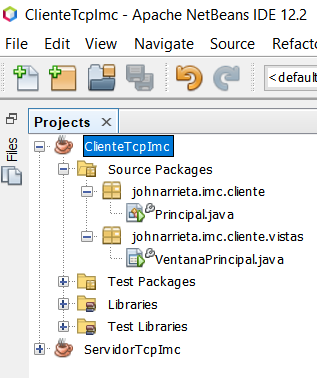
1. En la casilla de project Names; colocamos el nombre del proyecto
2. y 3. damos click en el botón Browse y encontramos la carpeta donde queremos guardar el proyecto.

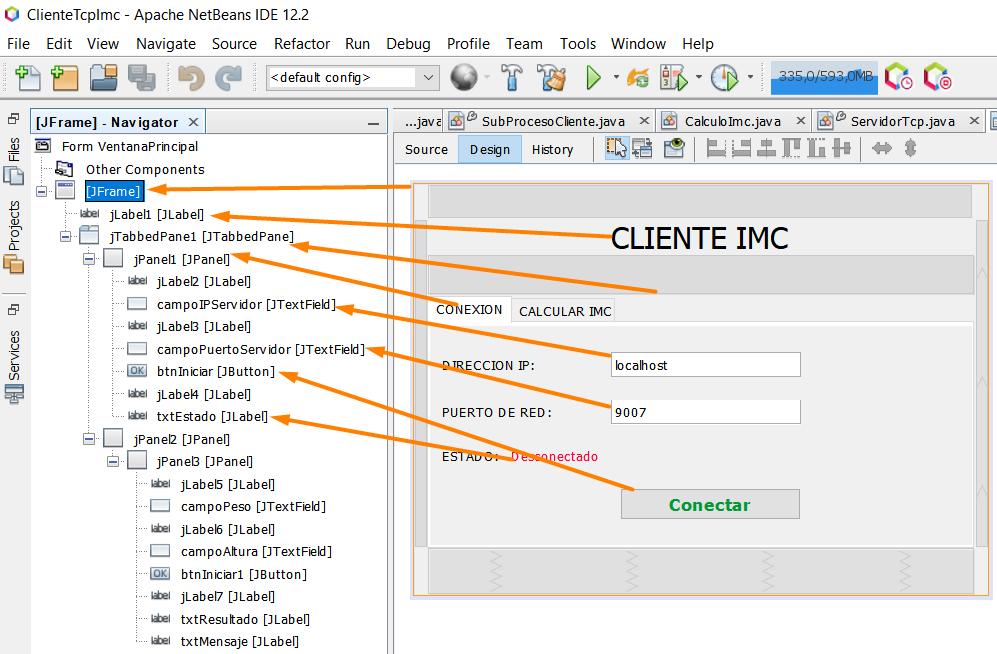
4. dar click en la casilla cuadrada y

5. al lado se coloca el nombre de la main class.

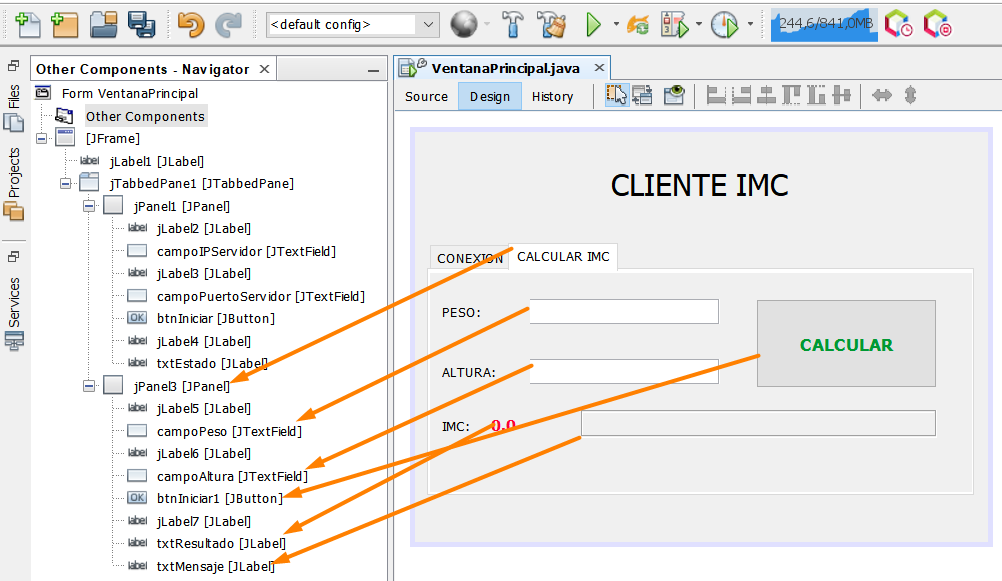
6. finalizamos todo dando click en el botón finish

Revisamos si todo está bien.

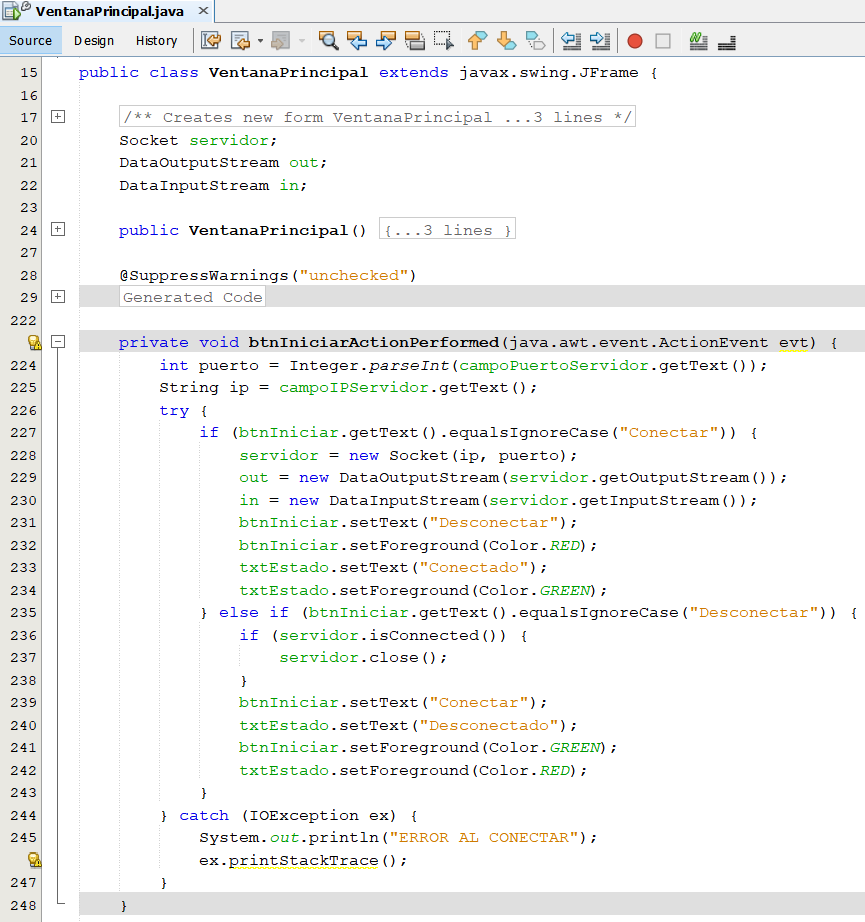




En el archivo ventana principal del cliente, creamos la ventana de la parte del cliente con un jtable creamos el nombre “cliente IMC”, debajo ponemos un tabbedpane; dentro de él ponemos 2 panel, en el primer panel llamado conexión ponemos dos textfield; uno para dirección ip y otro para el puerto (cada textfield tiene su respectivo jlabel) además dos jlabel; uno para el estado y otro para la respuesta del estado, al final con un botón con el nombre conectar.



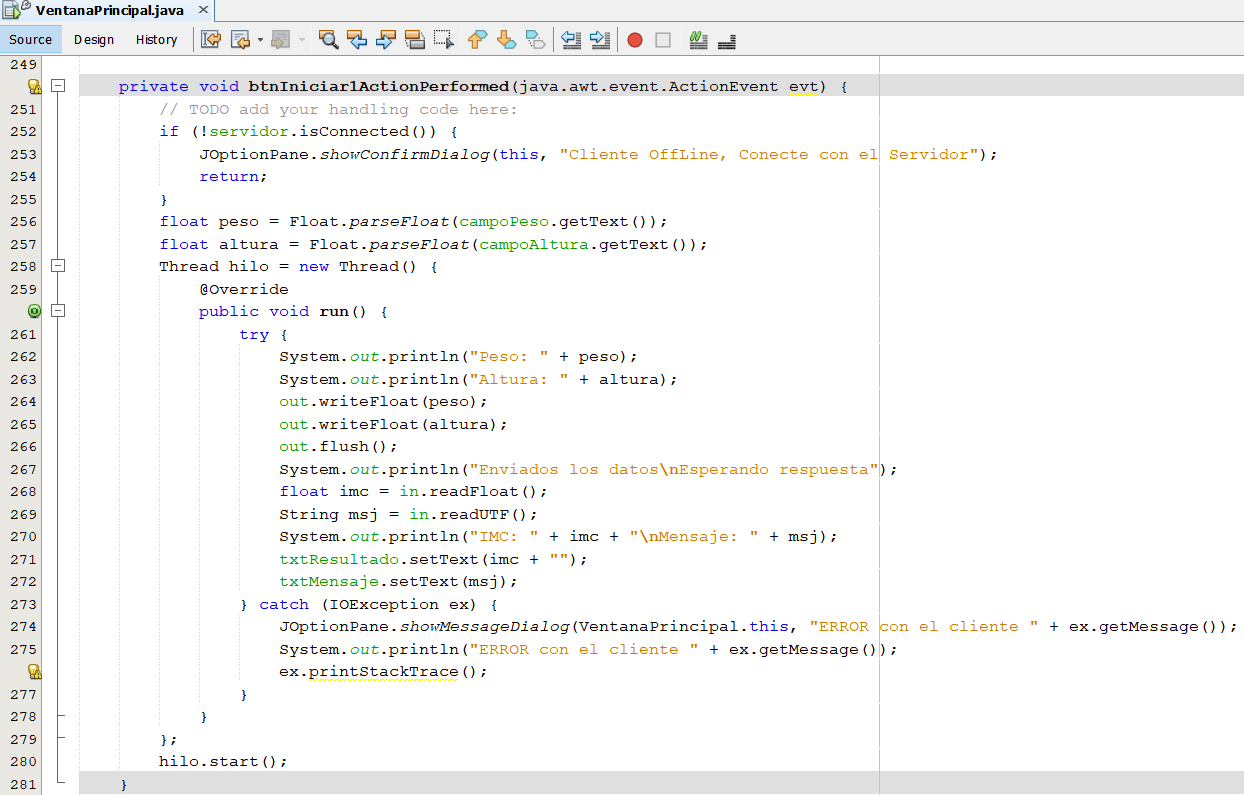
en el otro panel lo llamamos calcular imc, dentro ponemos un tabbedpane, en el le ponemos dos textfield; uno para peso y otro para el altura (cada textfield tiene su respectivo jlabel con su nombre) además dos jlabel; uno que diga IMC y otro para la respuesta del IMC, al final con un botón con el nombre calcular.

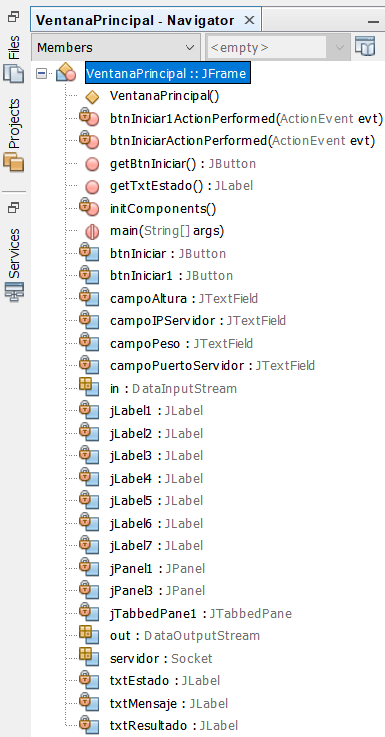


Se importan las clases de socket, dataoutputstream, datainputstream, se crea un atributo de cada clase.

Se crea una variable puerto que obtiene el puerto de la casilla campo puerto servidor, ip del servidor, después se crea un try catch; adentro se crea un if si el botón iniciar tiene el texto conectar entra al if, se instancia el objeto servidor tomando la dirección ip y el puerto.

si se desconecta el a través del botón, se cierra el servidor, si todo falla se envia el error.



Primero, el programa comprueba si un servidor está conectado. Si no lo está, muestra un mensaje al usuario para informarle de que el cliente está offline y debe conectarse con el dispositivo correspondiente. A continuación, el programa convierte el texto de entrada en variables de tipo float, lo que permite realizar operaciones matemáticas con estos valores. Se crea y ejecuta un nuevo hilo para realizar tareas en segundo plano. Este enfoque permite que el programa principal siga funcionando sin interrupciones, mejorando la eficiencia y la capacidad de respuesta. El programa también maneja la impresión y escritura de variables en la consola o en una salida externa. Esto es crucial para la visualización y registro de datos, así como para la interacción con otros sistemas o componentes. Finalmente, el programa incluye mecanismos para leer valores y manejar posibles errores de entrada/salida. 

verificamos que todo está bien.

| package johnarrieta.imc.cliente.vistas;  import java.awt.Color; import java.io.DataInputStream; import java.io.DataOutputStream; import java.io.IOException; import java.net.Socket; import javax.swing.JButton; import javax.swing.JLabel; import javax.swing.JOptionPane;  */\*\*  \* @author JOHN CARLOS ARRIETA ARRIETA  \*/* public class VentanaPrincipal extends javax.swing.JFrame {   */\*\*  \* Creates new form VentanaPrincipal  \*/*  Socket servidor;  DataOutputStream out;  DataInputStream in;   public VentanaPrincipal() {  initComponents();  }   @SuppressWarnings("unchecked")  *// <editor-fold defaultstate="collapsed" desc="Generated Code">*   private void initComponents() {   jLabel1 = new javax.swing.JLabel();  jTabbedPane1 = new javax.swing.JTabbedPane();  jPanel1 = new javax.swing.JPanel();  jLabel2 = new javax.swing.JLabel();  campoIPServidor = new javax.swing.JTextField();  jLabel3 = new javax.swing.JLabel();  campoPuertoServidor = new javax.swing.JTextField();  btnIniciar = new javax.swing.JButton();  jLabel4 = new javax.swing.JLabel();  txtEstado = new javax.swing.JLabel();  jPanel3 = new javax.swing.JPanel();  jLabel5 = new javax.swing.JLabel();  campoPeso = new javax.swing.JTextField();  jLabel6 = new javax.swing.JLabel();  campoAltura = new javax.swing.JTextField();  btnIniciar1 = new javax.swing.JButton();  jLabel7 = new javax.swing.JLabel();  txtResultado = new javax.swing.JLabel();  txtMensaje = new javax.swing.JLabel();   setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);   jLabel1.setFont(new java.awt.Font("Tahoma", 0, 24)); *// NOI18N*  jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);  jLabel1.setText("CLIENTE IMC");   jLabel2.setText("DIRECCION IP: ");   campoIPServidor.setText("localhost");   jLabel3.setText("PUERTO DE RED:");   campoPuertoServidor.setText("9007");   btnIniciar.setFont(new java.awt.Font("Tahoma", 1, 14)); *// NOI18N*  btnIniciar.setForeground(new java.awt.Color(0, 153, 51));  btnIniciar.setText("Conectar");  btnIniciar.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  btnIniciarActionPerformed(evt);  }  });   jLabel4.setText("ESTADO: ");   txtEstado.setForeground(new java.awt.Color(255, 0, 51));  txtEstado.setText("Desconectado");   javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);  jPanel1.setLayout(jPanel1Layout);  jPanel1Layout.setHorizontalGroup(  jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(jPanel1Layout.createSequentialGroup()  .addContainerGap()  .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)  .addGroup(jPanel1Layout.createSequentialGroup()  .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED\_SIZE, 131, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  .addComponent(campoIPServidor, javax.swing.GroupLayout.PREFERRED\_SIZE, 152, javax.swing.GroupLayout.PREFERRED\_SIZE))  .addGroup(jPanel1Layout.createSequentialGroup()  .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED\_SIZE, 131, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  .addComponent(campoPuertoServidor))  .addGroup(jPanel1Layout.createSequentialGroup()  .addComponent(jLabel4)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  .addComponent(txtEstado, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))  .addComponent(btnIniciar, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.PREFERRED\_SIZE, 145, javax.swing.GroupLayout.PREFERRED\_SIZE))  .addContainerGap(137, Short.MAX\_VALUE))  );  jPanel1Layout.setVerticalGroup(  jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(jPanel1Layout.createSequentialGroup()  .addGap(21, 21, 21)  .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  .addComponent(jLabel2)  .addComponent(campoIPServidor, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))  .addGap(18, 18, 18)  .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  .addComponent(jLabel3)  .addComponent(campoPuertoServidor, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))  .addGap(18, 18, 18)  .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  .addComponent(jLabel4)  .addComponent(txtEstado))  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 19, Short.MAX\_VALUE)  .addComponent(btnIniciar)  .addGap(20, 20, 20))  );   jTabbedPane1.addTab("CONEXION", jPanel1);   jLabel5.setText("PESO:");   jLabel6.setText("ALTURA:");   btnIniciar1.setFont(new java.awt.Font("Tahoma", 1, 14)); *// NOI18N*  btnIniciar1.setForeground(new java.awt.Color(0, 153, 51));  btnIniciar1.setText("CALCULAR");  btnIniciar1.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  btnIniciar1ActionPerformed(evt);  }  });   jLabel7.setText("IMC: ");   txtResultado.setFont(new java.awt.Font("Tahoma", 1, 12)); *// NOI18N*  txtResultado.setForeground(new java.awt.Color(255, 0, 51));  txtResultado.setText("0.0");   txtMensaje.setBorder(javax.swing.BorderFactory.createTitledBorder(""));   javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout(jPanel3);  jPanel3.setLayout(jPanel3Layout);  jPanel3Layout.setHorizontalGroup(  jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(jPanel3Layout.createSequentialGroup()  .addContainerGap()  .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(jPanel3Layout.createSequentialGroup()  .addComponent(jLabel7)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)  .addComponent(txtResultado, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  .addComponent(txtMensaje, javax.swing.GroupLayout.PREFERRED\_SIZE, 285, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addGap(0, 0, Short.MAX\_VALUE))  .addGroup(jPanel3Layout.createSequentialGroup()  .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(jPanel3Layout.createSequentialGroup()  .addComponent(jLabel6, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  .addComponent(campoAltura, javax.swing.GroupLayout.PREFERRED\_SIZE, 152, javax.swing.GroupLayout.PREFERRED\_SIZE))  .addGroup(jPanel3Layout.createSequentialGroup()  .addComponent(jLabel5, javax.swing.GroupLayout.PREFERRED\_SIZE, 66, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  .addComponent(campoPeso, javax.swing.GroupLayout.PREFERRED\_SIZE, 152, javax.swing.GroupLayout.PREFERRED\_SIZE)))  .addGap(29, 29, 29)  .addComponent(btnIniciar1, javax.swing.GroupLayout.PREFERRED\_SIZE, 145, javax.swing.GroupLayout.PREFERRED\_SIZE)))  .addGap(28, 28, 28))  );  jPanel3Layout.setVerticalGroup(  jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(jPanel3Layout.createSequentialGroup()  .addGap(21, 21, 21)  .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)  .addGroup(jPanel3Layout.createSequentialGroup()  .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  .addComponent(jLabel5)  .addComponent(campoPeso, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))  .addGap(28, 28, 28)  .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  .addComponent(jLabel6)  .addComponent(campoAltura, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))  .addGap(21, 21, 21))  .addGroup(jPanel3Layout.createSequentialGroup()  .addComponent(btnIniciar1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)  .addGap(18, 18, 18)))  .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  .addComponent(jLabel7)  .addComponent(txtResultado, javax.swing.GroupLayout.PREFERRED\_SIZE, 22, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(txtMensaje, javax.swing.GroupLayout.PREFERRED\_SIZE, 22, javax.swing.GroupLayout.PREFERRED\_SIZE))  .addContainerGap(43, Short.MAX\_VALUE))  );   jTabbedPane1.addTab("CALCULAR IMC", jPanel3);   javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());  getContentPane().setLayout(layout);  layout.setHorizontalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(layout.createSequentialGroup()  .addContainerGap()  .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 437, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addComponent(jTabbedPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))  .addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))  );  layout.setVerticalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(layout.createSequentialGroup()  .addGap(27, 27, 27)  .addComponent(jLabel1)  .addGap(32, 32, 32)  .addComponent(jTabbedPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 203, javax.swing.GroupLayout.PREFERRED\_SIZE)  .addContainerGap(37, Short.MAX\_VALUE))  );   pack();  }*// </editor-fold>*    private void btnIniciarActionPerformed(java.awt.event.ActionEvent evt) {   int puerto = Integer.parseInt(campoPuertoServidor.getText());  String ip = campoIPServidor.getText();  try {  if (btnIniciar.getText().equalsIgnoreCase("Conectar")) {  servidor = new Socket(ip, puerto);  out = new DataOutputStream(servidor.getOutputStream());  in = new DataInputStream(servidor.getInputStream());  btnIniciar.setText("Desconectar");  btnIniciar.setForeground(Color.RED);  txtEstado.setText("Conectado");  txtEstado.setForeground(Color.GREEN);  } else if (btnIniciar.getText().equalsIgnoreCase("Desconectar")) {  if (servidor.isConnected()) {  servidor.close();  }  btnIniciar.setText("Conectar");  txtEstado.setText("Desconectado");  btnIniciar.setForeground(Color.GREEN);  txtEstado.setForeground(Color.RED);  }  } catch (IOException ex) {  System.out.println("ERROR AL CONECTAR");  ex.printStackTrace();  }  }    private void btnIniciar1ActionPerformed(java.awt.event.ActionEvent evt) {   *// TODO add your handling code here:*  if (!servidor.isConnected()) {  JOptionPane.showConfirmDialog(this, "Cliente OffLine, Conecte con el Servidor");  return;  }  float peso = Float.parseFloat(campoPeso.getText());  float altura = Float.parseFloat(campoAltura.getText());  Thread hilo = new Thread() {  @Override  public void run() {  try {  System.out.println("Peso: " + peso);  System.out.println("Altura: " + altura);  out.writeFloat(peso);  out.writeFloat(altura);  out.flush();  System.out.println("Enviados los datos\nEsperando respuesta");  float imc = in.readFloat();  String msj = in.readUTF();  System.out.println("IMC: " + imc + "\nMensaje: " + msj);  txtResultado.setText(imc + "");  txtMensaje.setText(msj);  } catch (IOException ex) {  JOptionPane.showMessageDialog(VentanaPrincipal.this, "ERROR con el cliente " + ex.getMessage());  System.out.println("ERROR con el cliente " + ex.getMessage());  ex.printStackTrace();  }  }  };  hilo.start();  }    public JLabel getTxtEstado() {  return txtEstado;  }   public JButton getBtnIniciar() {  return btnIniciar;  }   */\*\*  \* @param args the command line arguments  \*/*  public static void main(String args[]) {  */\* Set the Nimbus look and feel \*/*  *//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">*  */\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.  \* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html   \*/*  try {  for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {  if ("Nimbus".equals(info.getName())) {  javax.swing.UIManager.setLookAndFeel(info.getClassName());  break;  }  }  } catch (ClassNotFoundException ex) {  java.util.logging.Logger.getLogger(VentanaPrincipal.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (InstantiationException ex) {  java.util.logging.Logger.getLogger(VentanaPrincipal.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (IllegalAccessException ex) {  java.util.logging.Logger.getLogger(VentanaPrincipal.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (javax.swing.UnsupportedLookAndFeelException ex) {  java.util.logging.Logger.getLogger(VentanaPrincipal.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  }  *//</editor-fold>*  *//</editor-fold>*   */\* Create and display the form \*/*  java.awt.EventQueue.invokeLater(new Runnable() {  public void run() {  new VentanaPrincipal().setVisible(true);  }  });  }   *// Variables declaration - do not modify*   private javax.swing.JButton btnIniciar;  private javax.swing.JButton btnIniciar1;  private javax.swing.JTextField campoAltura;  private javax.swing.JTextField campoIPServidor;  private javax.swing.JTextField campoPeso;  private javax.swing.JTextField campoPuertoServidor;  private javax.swing.JLabel jLabel1;  private javax.swing.JLabel jLabel2;  private javax.swing.JLabel jLabel3;  private javax.swing.JLabel jLabel4;  private javax.swing.JLabel jLabel5;  private javax.swing.JLabel jLabel6;  private javax.swing.JLabel jLabel7;  private javax.swing.JPanel jPanel1;  private javax.swing.JPanel jPanel3;  private javax.swing.JTabbedPane jTabbedPane1;  private javax.swing.JLabel txtEstado;  private javax.swing.JLabel txtMensaje;  private javax.swing.JLabel txtResultado;  *// End of variables declaration*  } |
| --- |

**ARCHIVO PRINCPAL.JAVA**

| package johnarrieta.imc.cliente;  import johnarrieta.imc.cliente.vistas.VentanaPrincipal;  */\*\*  \*  \* @author JOHN CARLOS ARRIETA ARRIETA  \*/* public class Principal {   */\*\*  \* @param args the command line arguments  \*/*  public static void main(String[] args) {  VentanaPrincipal v = new VentanaPrincipal();  v.setLocationRelativeTo(null);  v.setVisible(true);  }  } |
| --- |

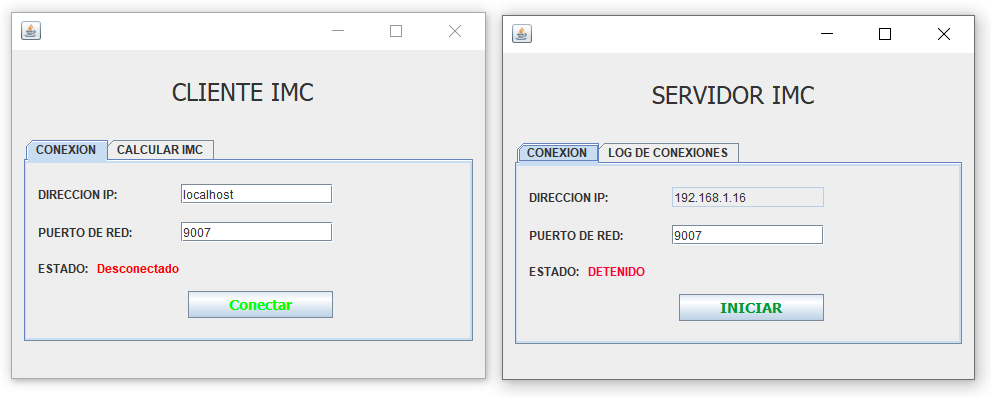
Se importan las clases de socket, dataoutputstream, datainputstream, se crea un atributo de cada clase.

Se crea una variable puerto que obtiene el puerto de la casilla campo puerto servidor, ip del servidor, después se crea un try catch; adentro se crea un if si el botón iniciar tiene el texto conectar entra al if, se instancia el objeto servidor tomando la dirección ip y el puerto.

si se desconecta el a través del botón, se cierra el servidor, si todo falla se envia el error.

Primero, el programa comprueba si un servidor está conectado. Si no lo está, muestra un mensaje al usuario para informarle de que el cliente está offline y debe conectarse con el dispositivo correspondiente. A continuación, el programa convierte el texto de entrada en variables de tipo float, lo que permite realizar operaciones matemáticas con estos valores. Se crea y ejecuta un nuevo hilo para realizar tareas en segundo plano. Este enfoque permite que el programa principal siga funcionando sin interrupciones, mejorando la eficiencia y la capacidad de respuesta. El programa también maneja la impresión y escritura de variables en la consola o en una salida externa. Esto es crucial para la visualización y registro de datos, así como para la interacción con otros sistemas o componentes. Finalmente, el programa incluye mecanismos para leer valores y manejar posibles errores de entrada/salida.

**LAS PRUEBAS**





probamos que la conexión funciona bien.

